AN ARGUS SPECIALIST PUBLICATION

NEW

MAY/JUNE 1983

A&B COMPUTING

£1.85

FOR THE USER OF THE BBC MICRO

Over 120 pages of features, programs, news and reviews to help you get more from your BBC Micro!

Reviews
We spotlight BBC software

Cells and Serpents
An exciting epic adventure to key in

Programs
Amazing new games and utilities for you to run

PLUS: Education - Get a grounding in graphics
Printers - Try Tandy's £150 alternative

FREE TO ENTER CONTEST
Win £100's worth of the latest software

Midwich
COMPUTER COMPANY LIMITED

1st choice for BBC microcomputers

BBC COMPUTERS

Model B		£346.95
Model B + Disc Interface		£441.95

BBC MICRO DISC DRIVES

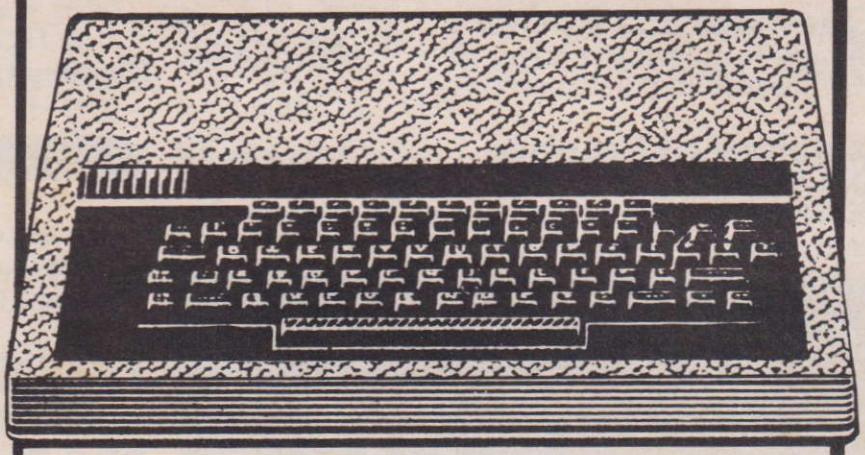
BBC 31	Single 100K Drive Expandable to 2 x 100K	£199.00
BBC 32	Dual 100K Drives	£330.00
BBC 33	100K Upgrade for BBC 31	£122.00
BBC 34	Dual 400K Drives	£649.00

BBC 35 Acorn Utilities Disc/Manual £30.00
(supplied only with BBC 31, 32, 34)
All Drives except BBC 33 supplied cased with Connecting Leads.

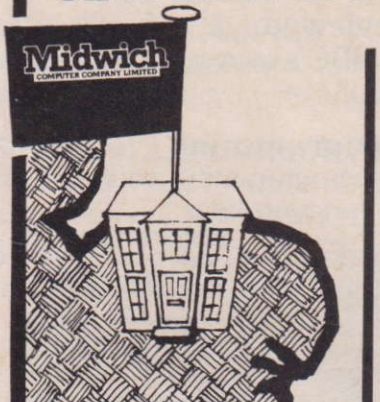
BBC UPGRADE KITS

BBCA2B	Complete A to B Upgrade	£44.75
BBC 1	16K Memory	£18.00
BBC 2	Printer/User 1/0 Kit	£ 7.50
BBC 3	Disc Interface Kit	£95.00
BBC 4	Analogue Input Kit	£ 6.70
BBC 5	Serial 1/0 RGB Kit	£ 7.30
BBC 6	Bus. Expansion Kit	£ 6.45

All kits are supplied with full fitting instructions.



Fast ex-stock delivery by Securicor...



Fitting Service available

12 month "no quibble" warranty on all products

BBC CONNECTORS

BBC 21	Printer Cable and Amphenol Plug (not assembled)	£13.00
BBC 22	User Port Connector and Cable 36"	£ 2.46
BBC 23	Cassette Lead	£ 3.50
BBC 24	7 Pin Din Plug	£ 0.60
BBC 25	6 Pin Din Plug	£ 0.60
BBC 26	5 Pin Din Plug	£ 0.60
BBC 35	Disc 1/0 Cable 34W IDC to 2 x 34 way Card Edge	£12.00
BBC 36	Disc Power Cable	£ 6.00
BBC 44	Analogue Input Plug & Lever	£ 2.25
BBC 66	1 M Bus Connector + 36" Cable	£ 3.50

BBC ACCESSORIES

BBC 45	Joysticks (per pair)	£11.30
BBC 67	Eprom Programmer (assembled)	£57.95

ACORNSOFT FOR THE BBC

SBE03	Business Games	£ 8.65
SBE04	Tree of Knowledge	£ 8.65
SBE02	Peeko Computer Inc Manual	£ 8.65
SBE01	Algebrail Manipulation PK	£ 8.65
SBX01	Creative Graphics Cassette	£ 8.65
SBX02	Graphs & Charts Cassette	£ 8.65
SBB01	Desk Diary Inc Manual	£ 8.65
SBL02	Lisp Cassette	£14.65
SBL01	Forth Cassette	£14.65
SBG01	Philosophers Quest	£ 8.65
SBG07	Sphinx Adventure	£ 8.65
SBG03	Monsters	£ 8.65
SBG04	Snapper	£ 8.65
SBG15	Planetoid	£ 8.65
SBG06	Arcade Action	£10.35
SBG05	Rocket Raid	£ 8.65
SBG13	Meteors	£ 8.65
SBG14	Arcadians	£ 8.65
SBG10	Chess	£ 8.65

ACORNSOFT BOOKS FOR THE BBC MICRO

SBD01	Creative Graphics	£ 7.50
SBD02	Graphs - Charts	£ 7.50
SBD04	Lisp	£ 7.50
SBD03	Forth	£ 7.50

* Please ring for current delivery on Acornsoft products before ordering.

BBC MICRO COMPONENTS

4516 100NS	£ 2.25
6522	£ 3.19
74LS244	£ 0.59
74LS245	£ 0.69
74LS163	£ 0.34
DS3691N	£ 4.50
DS88LS120N	£ 4.50
UPD7002	£ 4.50
8271	£36.00
20 Way Header	£ 1.46
26 Way Header	£ 1.76
34 Way Header	£ 2.06
40 Way Header	£ 2.32
15 Way D Skt	£ 2.15
6 Way Din Skt	£ 0.90
5 Way Din Skt	£ 0.90

BBC SOFTWARE IN ROM.

Wordprocessor "View"	£52.00
1.2 MOS	£10.00

...from East Anglias leading supplier

Delivery Charges
Computers/Disc Drives £5.00
Components/Software £0.50
Books/Joysticks £1.00

MIDWICH COMPUTER COMPANY LIMITED

RICKINGHALL HOUSE RICKINGHALL SUFFOLK IP22 1HH
TELEPHONE (0379) DISS 898751

A&B Computing is constantly on the look out for well-written articles and programs for publication. If you feel that your efforts meet our standards, please feel free to submit your work to us for consideration.

All submitted material should be typed, double spaced if possible, and any program submitted should be listed, a cassette of the program alone will not be considered. All programs must come complete with a full explanation of the operation and, where relevant, the structure; cassettes of the program should also be included so that screen photographs and printer dumps can be included to illustrate the article. (Cassettes will, of course, be returned in due course).

All submissions will be acknowledged and any published work will be paid for at competitive rates. All work for consideration should be sent to the Editor of A&B Computing at our Charing Cross Road address.

Volume One Number

Editor: Elspeth Joiner
Deputy Editor: Roger Munford
Consultant Editor: Henry Budgett
Advertising Manager: Miriam Roberts
Divisional Advertising Manager: Beverley McNeill
Managing Editor: Ron Harris

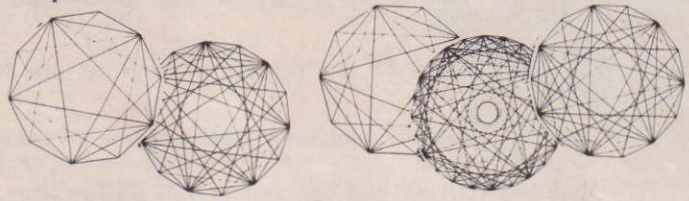
&B

One May/June 1983

Managing Director: TJ Connell

Origination and design by MM Design & Print, 145 Charing Cross Road, London WC2H 0EE.

Published by Argus Specialist Publications Ltd, 145 Charing Cross Road, London WC2H 0EE.



A&B Computing is published bi-monthly on the fourth friday of the month. Distributed by: Argus Press Sales & Distribution Ltd, 12-18 Paul Street, London EC2A 4JS. Telephone: 01-247 8233. Printed by Henry Garnett Ltd, Rotherham.

FEELING ADVENTUROUS?

ASP Software Adventure Series 1

THE WHITE BARROWS Program approximately 8K
Somewhere amid this maze of burial chambers lurks an Evil Sorcerer whom you need to trap. Trouble is, he's protected by Trolls, Dwarves, Serpents and the occasional Dragon or two! Your magic staff will block the tunnel to prevent him escaping unless, that is, he outwits you.

A real brain twister, White Barrows requires both brains and brawn from its players. It's no good just hacking your way through the Barrows and hoping to fall over the Sorcerer. Eventually you'll meet a Dragon, and they don't hack easily! You'll need all your strength and cunning to survive this one for long.

THE WHITE BARROWS Only £6.50 all inclusive!

CONQUERING EVEREST Program approximately 11K
You are in charge of an expedition comprising 18 climbers, 34 Sherpas and 40 Porters. There is food, tents and equipment for all, even the oxygen you'll need as you near the summit. One slight problem, it's all at the BOTTOM of the mountain and you have to get it all up to the TOP!

The monsters of this game are avalanches, starvation, storms and, worst of all, bad planning! A real, thinking man's adventure, Everest will test your skills of forward planning to the limit.

CONQUERING EVEREST Only £6.50 all inclusive!

** SPECIAL DEAL ** Both programs on one tape for only £11.45 all inclusive!

ASP Software Adventure Series 2

CELLS AND SERPENTS Program approximately 11K
More monsters than you ever thought could live behind your keyboard. Wander the hills in search of gold and glory but be very, very careful where you tread! There are things here that will make your wildest nightmares look like Julie Andrews. Fancy meeting a Mind Flyer, for example? Or how about shaking hands with an Asmodeus? (You'll only do that once!) Treasure is here to be found though...the hard way.

See just how good you really are at adventuring with this practically unsurvivable fantasy. Not for the faint of heart or the slow of sword.

CELLS AND SERPENTS Only £6.50 all inclusive!

STOCKMARKET Program approximately 11K
There are other ways of making money than bashing Trolls on the head. Try this one for a change. Contend with a fluctuating economy, tax investigations, bullish opponents, impatient bank managers and consortium takeovers as you struggle to make your first million.

It is decidedly difficult and definitely compulsive. A must for all those aspiring financial wizards, both young and old, the game has real family appeal as up to six people can play. It's easy to learn but very, very hard to win!

STOCKMARKET Only £6.50 all inclusive!

** SPECIAL DEAL ** Both programs on one tape for only £11.45 all inclusive!

Our Adventure Series programs are available on tape for the following systems:
Commodore VIC-20 (not available for White Barrows/Everest), Sharp MZ-80A and MZ-80K, Tandy TRS-80 Model 1, BBC Model B or 32K Model A, Atari 400 and 800, Sinclair 48K ZX Spectrum.

Please indicate your selection in the coupon and give the relevant system. Please note that we CANNOT supply mixed orders on the combined tapes; A BBC Cells and Serpents with an Atari Stockmarket, for example.

Fill in the coupon, cut it out and send it to:

ASP Software, ASP Ltd, 145 Charing Cross Road, London WC2H 0EE

Please send me ..... tape(s) of the following programs:

ASP Software Adventure Series 1
The White Barrows . . . . . .@ £6.50 each
Conquering Everest . . . . . .@ £6.50 each
Combined tape . . . . . @ £11.45

ASP Software Adventure Series 2
Cells and Serpents . . . . . .@ £6.50 each
Stockmarket. . . . . .@ £6.50 each
Combined tape . . . . . @ £11.45

My system is a . . . . . computer

I am enclosing my Cheque/Postal Order/Money Order (delete as necessary) for £ . . . .(payable to ASP Ltd)
OR
Debit my Access/Barclaycard (delete as necessary)

Please use BLOCK CAPITALS and include your postcode
NAME (Mr/Mrs/Miss). . . . . .
ADDRESS . . . . . .
. . . . . .
. . . . . .POSTCODE. . . . . .
Signature . . . . . .Date . . . . . .

Welcome Welcome

First off, I would like to say a big 'hello' to everyone who reads this launch issue of *A & B Computing*.

And now let's get down to the nitty gritty of what you'll find within *A & B Computing*. There are no prizes for guessing just who the magazine will appeal to—yes, all those who own a BBC Micro be it an A or B model.

There's something for everyone, right from the person who has never before clapped eyes on a computer to those who have written more programs than eaten hot dinners. Beginners will find our special series 'In the Beginning' invaluable help during their early programming days. Computer novices can also glean a lot from our article 'Bar Charts for Beginners'. BBC Micro owners who want to dig more deeply will find all they need to know in articles about the operating system and about the disc drive system.

Keen gamesters will have a great deal to get to grips with in our mammoth adventure game 'Cells and Serpents'. This was originally produced by ASP Software and has been completely re-vamped to improve it as a game and to use the best facilities of the BBC Micro.

Regular features will include software review of games—educational, domestic and utility programs. So, any firms interested in *A & B Computing* reviewing their products just send in software and instructions to us at our Charing Cross Road address.

Each issue you will get a chance to air your views about topics close to your heart and the BBC Micro's. There'll be pages where our technical expert will answer as many queries as possible relating to the BBC Micro. If you have the slightest problem with programming your machine don't hesitate to let us know about it.

We are always on the lookout for good programs and articles for future issues of *A & B Computing*, and where better to look than to our own readers. If, when reading through the magazine, you think you can write programs as well, or better than, our present contributors, then lets hear from you.

All contributions are, of course, paid for at very competitive rates. So, if you've got your eye on a new BBC add-on or you'd just like to supplement your pocket money, get writing! It is vital, though, that all the programs you send to us are totally original, and not 'borrowed' or 'adapted' from other magazines or books.

Any kind of program (business, domestic, educational, or just fun) will be welcomed, but particularly those which use BBC BASIC in clever and efficient ways, or those which employ certain routines which can be used in other programs.

Program listings are vital, along with a clear explanation of how the program is constructed, what it does and what the user can expect to see once the program is RUN (a screen dump is particularly valuable in this respect). When submitting your programs, it is very important to remember to enclose a cassette of the program as well as the listing, as this will allow us to check the program before publication.

Elspeth Joiner,
Editor.

Questions & Answers

J Ruston

Each issue our resident expert, Jeremy Ruston, will answer a selection of technical queries which are causing you problems.

Q. My BBC Micro has recently been upgraded with a disc interface. I hope to get some drives soon, but the interface came without a manual, so I don't know how. Also, I can't see how to use my old cassettes. I hope you can help me over these matters.

A. *To answer your second query first, to use your old cassettes you must do two things. The first is to tell the computer to load programs from cassette rather than your (non-existent) disc drive. This can be done in two ways. The easiest way is to keep the Space Bar pressed down while you press Break. This will initialise the 1200 baud cassetted system (in the same way, pressing 'D' whilst doing a Break initialised the disc filing system, and 'N' intialised the Econet filing system). If you want to use 300 baud cassettes, type '*TAPE3'. Once the computer has been instructed to use the cassette filing system, you must tell it to reclaim the area of memory normally used by the disc filing system by typing 'PAGE=&E00'. This releases the normal cassette complement of memory. These two steps are all that is required. (It is worth pointing out that the disc interface 'steals' 3K of memory from the user, by giving PAGE a default value of &1900 Hex. A useful fact, omitted from the disc* **User Guide,** *is that PAGE can be lowered to &1100 Hex quite safely, provided all you do with the disc system is SAVE and LOAD programs—in other words, if you use data files, you must leave PAGE at &1900 Hex.)*

The disc interface does not come with a manual for the rather dubious reason than 'non-BBC drives cannot be guaranteed to work with the BBC Microcomputer'. To get drives fitted to your computer you must contact a dealer. Any dealer will fit the drives for you, and many of them sell drives much cheaper than the 'official' BBC drives. There is another reason for doing this—because you did not buy the official drives, you were not supplied with the BBC Utilities disc, which contains the vital program to enable you to format discs, without which the system is unusable. However some dealers might be prepared to let you have one of these formatting programs. Alternatively, The Computer Users Club has its own utilities disc, which contains their own formatting program, which closely resembles the Acorn version. (The other programs on the disc are not of the highest standard, but are worth having.)

Q. I have been having a great deal of trouble with the BBC assembler. A section of code looking like this in one of my programs refuses to work:

```
.
.
.
ACC=&80
.
.
.
ASL ACC
.
.
.
```

A. *The reason this will not work is due to an obscure 'feature' of the assembler. The assembler instruction ASL operates in a number of addressing notes. One of these is Accumulator addressing, as in 'ASL A', which logically shifts left the Accumulator. However, the assembler reads ALL 'ASL' instructions followed by the letter 'A' as being 'ASL A', and assumes anything following the 'A' is a comment. Thus, your example becomes, in the assembler's mind, 'ASL A / CC'. The solution to this is to use 'ASL (ACC)'.*

This way of commenting assembler programs, without using the backlash, is actually rather neat, but being undocumented, it has led to a number of problems, like the above.

Q. I am very confused by the plethora of different ROMs and EPROMs fitted to BBC Micros. Exactly what are the differences between the operating systems and between BASIC I and BASIC II? I am sure there must be others who are similarly concerned.

A. *The different operating systems have caused the most problems. When the BBC Micro first came out, what was effectively a provisional operating system was supplied. This was known as 'OS EPROM 0.10', soon to be followed by the same thing in ROM. Version 0.10 was really pretty terrible (but Acorn are taking an odd line on this, and will not automatically replace it for those stuck with it). Its major defects were a number of weird bugs and the inability to cope with plug-in ROMs, speech, discs or the RS423 interface. It was followed by a number of versions like 0.87, 0.49 and so on, which never achieved wide circulation, but which are nevertheless present. These were capable of supporting most peripherals, but were somewhat bug-ridden. The major breakthrough came with version 1.0, which was sent out with the early disc systems (in EPROMs). Version 1.1 followed, but only small numbers made their way to the public. Version 1.2 is the new final version, and is now available in ROM. Most users will not notice what operating system they use, but it is worth obtaining 1.2, since many of the more serious software houses are beginning to issue programs which require its use.*

The disc filing system has also appeared in a number of versions, but the differences are not significant.

The first version of BASIC sent out with the machines has been designated issue I. It was more or less perfect, but contained a number of silly bugs:

* *ELSE did not work in ON...GOTO/GOSUB statements.*
* *INSTR where the second argument is longer than the first caused chaos.*
* *Statements such as 'EVAL ("TIME")' did not work.*
* *ABS with a positive integer argument often returned a string.*
* *Changing MODE did not reset COUNT.*

There are others, but they only arise in odd circumstances. BASIC II avoides these bugs, and adds one new command and some new assembler features. The new command is OSCLI, which takes a string argument, and passes it to the operating system. This allows you to use statements such as OSCLI "KEY "+STR$(T%)+ " hello", ie commands normally used with the asterisk prefix can

be written in such a way that the parameters of the OS commands are not known at run time. (Both BASICs provide an undocumented command which is very useful when using OSCLI, or its BASIC I equivalent (see **User Guide** p. 463). This is STR$ (X%), which returns a string of hexadecimal characters, equivalent to X%. Thus, one can write 'OSCLI "LOAD "+ FILE$+" "+STR$ (SPACE%)'.

The new assembler features are even more useful. EQUB, EQUD, EQUW and EQUS can be used to insert bytes, double bytes, 32-bit words and strings directly into assembly language programs. In addition, OPT now has a range from 0–7, the top bit allowing the code to be put at 0%, rather than P%. This makes it possible to assemble code at a place other than where it will be executed.

The new BASIC really deserves an article to itself!

Q. The picture of my BBC Micro flickers terribly when in MODES 0–6. Will this damage my Contec TV?

A. No damage will result, but you may still like to eliminate the flicker by typing '*TV 0,1' before you enter any of the above modes.

Q. Would you recommend VIEW over WORDWISE for simple word processing?

A. VIEW and WORDWISE are (currently) the only proper word-processing programs available for the BBC Micro. Both come in ROM, and both require operating system 1.0 or greater. There, the similarities end. WORDWISE is 8K long, while VIEW is a full 16K. Thus, VIEW is definitely more complex and more capable. Given that, there are still a lot of factors to consider when deciding between them. Being simpler, WORDWISE is probably easier to use. Thus, you must offset capabilities with ease of use, then the decision is up to you. If you have ever used WordStar on CP/M computers, you will feel at home with VIEW, because VIEW can be viewed (?) as a usable version of the same. As a point of interest, I am using WORDWISE at the moment, and experiencing no problems. I use WORDWISE for all writing.

Q. Which printer should I buy to use with my BBC Micro?

A. I am not in a position to say 'rush out and buy XYZ printer', for the simple reason that I only have experience of a limited number of printers. Most people seem to use the Epson MX-80—myself included—which is a reliable and versatile workhorse. The NEC range has fervent supporters, and it appears to be a good buy. The Seikosha range is cheap but the typeface does not appear very legible. A last minute contender is the Olivetti ink jet printer, marketted by Acorn (Olivetti also make the BBC single disc drive). The ink jet is as quiet as you could wish and handles high-resolution graphics as competently as the others, but the typeface seems to let it down a little. Long term reliability has yet to be seen, but I should look closely at it.

Q. Why is there no 'WHILE-WEND' construction in BBC BASIC?

A. I can only guess at the answer to this question, but the following explanation seems as likely as any other.

First, I had better describe what 'WHILE-WEND' means. The 'WHILE-WEND' construction is part of the armoury of Pascal, and as such is expected in structured BASIC. It is similar to 'REPEAT-UNTIL' except that the terminating condition follows the word 'WHILE' and thus appears at the top of the loop, rather than the bottom. The action of the loop is that when the word 'WHILE' is encountered, the expression following it is evaluated. If the condition is TRUE, the program continues in the normal way, but remembers where the word 'WHILE' appeared and jumps back to this spot when the word 'WEND' is encountered. If the condition is FALSE, it jumps over the body of the loop, to the statement after the word 'WEND'. Thus, the construction carries out code until a condition is met, but by virtue of having the condition at the start of the loop, the entire contents of the loop can be skipped if required. This makes it very useful in a number of circumstances.

To hazard a guess at why it does not appear in BBC BASIC (even though it was mentioned in the early language specifications of 1981), I think that the interpreter would have problems when situations like the following arose:

```
WHILE condition
.
IF another condition THEN
  WEND
.
.
.
WEND
```

In these circumstances, if the initial condition is false, which 'WEND' statement does the computer jump to? The obvious solution is to jump to the start of the IF statement, but if some of the variables in the 'IF' statement's condition are defined between the 'WHILE' and the 'IF' statement, a 'No such variable' message would have to be issued. All this would lead to the situation where 'WHILE' would only work some of the time in the way the programmer intended, which would be mightily confusing. I imagine that Acorn decided that to avoid the confusion, the construction should be left out entirely, which would appear to be a wise decision.

BBC Clubhouse Organiser

J Waterhouse

A useful program for beginners and is a good teaching aid for learning the intricacies of file handling techniques.

The program was initially designed to assist in the running of a computer studies evening class for the local authority. It is in its present form for use in clubs and designed for use on BBC Model 'B' connected by the parallel printer interface to an Oki Microline 80.

The program has two main aims:

1. To allow those with little or no experience of computers to gain hands on experience by guiding them through the routines to input data to the running program and then to receive a hard copy output of their personal entries.
2. To find out the needs of individuals for future meetings and to provide hard copy printout of this.

The program has additional advantages for teaching and demonstration purposes in that it uses file handling techniques, has room for considerable modification and expansion, and uses high resolution graphics to create charts and graphs.

The program is modular in construction and individual elements may be taken for use in isolation.

WHAT'S ON THE MENU

The program is written in BBC BASIC as a series of procedures linked together and called from a central program section which performs 'menu' selection.

When originally written, the program was in two distinct sections, one to gather data, deal with printer routines and save all the information gathered to tape; and the second to reload the information from the tape file and create the graphic displays. This method allows the use of the highest level graphic modes for the charts. As the program stands now, Mode 5 is used which gives a slightly chunky display for the piechart.

Use is made of the BBC Micro's file handling facilities; but a word of warning for those with the 0.1 Operating System, a suitable machine code program should be run before use to cure the 'bugs'. Those people with a 1.0 Operating System or higher should have no difficulty with this.

PROGRAM DESCRIPTION

Lines	Description
10 - 50	Set up TV picture, turn off auto repeat and cursor keys, and select printer and screen mode.
60 - 90	Initialize variables and READ DATA into array.
130 - 200	Print out main menu and accept choice using GET.
210 - 460	Select procedure to call using ON . . . GOSUB, clears screen and goes back to menu on RETURN. Incorporates secondary menu for graphs and sets Mode (this must be done outside of any procedure).
500 - 970	PROCname. Main procedure for input of data runs in a REPEAT UNTIL loop from line 520 to 950. The loop is either terminated by running out of space in the arrays or by entering a 'II' character at line 1200; this is checked for at line 550.
1020 - 1210	PROCdisplay. The above routine calls this to create a display between entries showing the Club's name or other such information. It is written in Mode 7 and the CHR$( ) may be replaced by embedded control codes with the double quotes. These may be obtained by suitable programming of the function keys. The codes, however, come out as graphic symbols when listed to the printer so have been replaced with the more normal CHR$( ).
1270 - 1350	PROCconvtoUC. Is used to convert all characters input to upper case so that later printout is neat. Is also essential if at a later stage a sort procedure is added to the program. This routine works by checking each character in turn from a temporary variable S and subtracting 32 if the ASCII code is a lower case letter. Numbers and other characters are ignored.
1390 - 1910	PROCquest Contains screen display for the questionnaire to discover people's interests. The questions are presented as a list of eight items and the screen is formatted to receive the answers at line 1750. Answers are stored in a temporary array until verified by the operator.
1960 - 2090	PROCmachine Second part of the questionnaire to determine which computer, if any, is owned and then prints out list from DATA statements. All information is stored in an array.
2140 - 2470	PROCoutput Print out a list of all the information input by one person and invites them to take a copy from the printer (for the best effect use a twin-ply roll in the printer). VDU1 is used in all the printer routines to send a character to the printer. Various numbers are sent to set up the number of characters per line and the size of line. VDU1, 10 is a line feed.
2520 -2710	PROCsave. Store all data onto a casette file; the variable

'count' is also stored so that the file may be reloaded and added to at a later stage.

2760 - 2930 PROCload
Load the cassettee file.

2980 - 3160 PROCprint
Print out a condensed list of all the members names together with their interests; ' %' is a variable used to set up the print fields and to formulate the manner in which numbers are printed out.

3210 - 3280 PROCend
Terminate RUN having first reset computer into its normal mode.

3330 - 3710 PROCpiechart.
Set up the four colours in Mode 5 at line 3340 and then works its way through the array of interests adding each element in turn into a running total 'sect( )'. Line 3430 prevents division by zero—thus, there will be no problems if all inputs should be zero. Line 3490 calculates a small angle dependent upon the total number input. Lines 3460 to 3490 calculate the enclosed angle of each segment of the chart and lines 3560 and 3570 set the halfway and end of segment markers. The chart is plotted in lines 3610 - 3670 and at the halfway stage of each segment the segment number is printed. The print cursor having been tied to the graphics cursor using VDU 5. Line 3690 prints out the heading and line 3700 waits for the Space Bar to be pressed before returning to the main menu.

3760 - 4200 PROCbarchart.
Set up colours in line 3780, totals each column of the array in lines 3800 to 3850 and calculates the percentage in lines 3870 to 3890. The routine also draws and labels the Y-axis in lines 3930 to 4040 and the X-axis in lines 4060 to 4100. The bars of the chart are plotted in lines 4120 to 4170 using PROCbox.

4240 - 4310 PROCbox.
Calculate the co-ordinates of each corner of the bar and draws using two triangles (PLOT85,x,y). H is the scaling factor.

The program uses 'meaningful' variable names in most places to make reading the program easier.

Storage of data is in the main confined to the two arrays, RECORD$, which has 50 elements for storing names, and RECORD, which has 50 by eight elements for storing all the other information.

EXPANSION

The first logical expansion would be to include addresses in the information gathered so that you could separate the display routines to make use of the higher resolution graphics. The advantage of this would be that the piechart in particular could then be fully anoted.

VARIABLES

Variable	Function
RECORD$( )	Storage of names.
RECORD (,)	Storage of numbers.
inter ( )	Temporary array.
mach$( )	Temporary array.
sect ( )	Temporary array
count	Array subscript.
G and G$	Variable used for 'GET'.
I	General loop counter.
J	General loop counter.
TEMP$ and TEMP2$	Both temporary storage for names.
L	Intermediate storage when converting from lower to upper case
F	Used to hold channel number for cassette files.
R	Radius
x and y	Co-ordinates of screen centre.
inc	Increment
Z	Angle at centre of piechart.
col and g	Colour to be plotted next.
anh and tot	Intermediates in calculating size of each segment in the piechart.
p and prin	Used to locate printing position.
X, Y and H	Plotting co-ordinates.

PROGRAM LISTING

```
  10 *TV255,1
  20 *FX11,0
  30 *FX4,1
  40 *FX5,1
  50 MODE7
  60 DIM RECORD$(50),RECORD(50,8)
  70 DIMinter(8),mach$(9),sect(8)
  80 count=1
  90 FOR I=0 TO 9:READ mach$(I):NEXT
 100 REM
 110 REM Selection of routines
 120 REM
 130 PRINTTAB(2,4)CHR$(129);"1.";CHR$(134);"ENTER
NAMES"
 140 PRINTTAB(2,6)CHR$(129);"2.";CHR$(134);"DISPL
AY GRAPH"
 150 PRINTTAB(2,8)CHR$(129);"3.";CHR$(134);"SAVE
TO TAPE"
 160 PRINTTAB(2,10)CHR$(129);"4.";CHR$(134);"LOAD
FROM TAPE"
 170 PRINTTAB(2,12)CHR$(129);"5.";CHR$(134);"LIST
TO PRINTER"
 180 PRINTTAB(2,14)CHR$(129);"6.";CHR$(131);"END
RUN"
 190 PRINTTAB(2,18)CHR$(129);"ENTER CHOICE";CHR$(
132);"1-6";
 200 G=GET:IF G>54 OR G<49 THEN190
 210 G=G-48
```

CONTINUED OVER

PROGRAM LISTING

```
  220 ON G GOSUB 250,300,390,410,430,450
  230 CLS
  240 GOTO130
  250 PROCname
  260 RETURN
  270 REM
  280 REM Secondary menu selects graphs
  290 REM
  300 CLS:@%=0:IF count=1 count=2
  310 PRINTTAB(10,8)"WHICH DISPLAY ?"
  320 PRINTTAB(10,10)"1. PIECHART"
  330 PRINTTAB(10,12)"2. BARCHART"
  340 PRINTTAB(10,15)"Enter 1 or 2";
  350 G=GET:IFG<>49 AND G<>50 THEN 340
  360 IF G=49:MODE5:PROCpiechart
  370 IF G=50:MODE5:PROCbarchart
  380 MODE7:RETURN
  390 PROCsave
  400 RETURN
  410 PROCload
  420 RETURN
  430 PROCprint
  440 RETURN
  450 PROCend
  460 RETURN
  470 REM
  480 REM Main routine for data input
  490 REM
  500 DEFPROCname
  510 CLS
  520 REPEAT
>
  530 VDU26
  540 PROCdisplay
  550 IF G=124 THEN 950
  560 VDU26
  570 CLS
  580 PRINTTAB(2,4)CHR$(134);"We have to maintain
a club register,"
  590 PRINTTAB(2)CHR$(134);"to help us do this ple
ase enter your"
  600 PRINTTAB(2)CHR$(134);"name where shown below
."
  610 PRINT
  620 PRINTTAB(2)CHR$(134);"You may correct an err
or by using"
  630 PRINTTAB(2)CHR$(134);"the key marked ";CHR$(
129);" DELETE."
  640 PRINTTAB(2)CHR$(134);"When each part of your
 name is"
  650 PRINTTAB(2)CHR$(134);"complete, press the la
rge key"
  660 PRINTTAB(2)CHR$(134);"marked ";CHR$(129);" R
ETURN."
  670 FOR I=1 TO 7000:NEXTI
  680 PRINT:PRINT
  690 PRINTTAB(6)CHR$(131);" TYPE YOUR";CHR$(129);
"FIRST";CHR$(131);"NAME"
  700 VDU28,3,24,39,16
  710 INPUT TEMP$
  720 IF VAL(TEMP$)<>0 THEN PRINT"PLEASE ENTER ALP
HABETIC CHARACTERS":FOR T=1 TO 3000:NEXT:CLS:GOTO7
10
  730 VDU28,3,24,39,19
  740 CLS
  750 PRINTTAB(4)CHR$(131);"TYPE YOUR";CHR$(129);"
LAST";CHR$(131);"NAME"
  760 VDU28,3,24,39,21
  770 INPUT TEMP2$
  780 IF VAL(TEMP2$)<>0 THEN PRINT"PLEASE ENTER AL
PHABETIC CHARACTERS":FOR T=1 TO 3000:NEXT:CLS:GOTO
770
  790 VDU28,2,24,39,14
  800 CLS
  810 TEMP$=TEMP$+" "+TEMP2$
  820 PROCconvtoUC
  830 PRINTTAB(0,3)CHR$(134);"Is your correct name
"
  840 PRINT
  850 PRINT"    ";TEMP$
  860 PRINT
  870 PRINTCHR$(134);"If this is correct type";CHR
$(130);"Y"
  880 PRINTCHR$(134);"If this is wrong type ";CHR$
(130);"N"
  890 G$=GET$:IFG$="N" OR G$="n":CLS:GOTO690
  900 IF G$="Y" OR G$="y" THEN RECORD$(count)=TEMP
$:ELSE CLS:GOTO830
  910 PROCquest
  920 PROCmachine
  930 PROCoutput
  940 count=count+1
  950 UNTIL G=124 OR count=50
  960 IF count=50 PRINT" FILE FULL, SAVE TO TAPE"
  970 ENDPROC
  980 REM
  990 REM Displays title page between
 1000 REM different inputs
 1010 REM
 1020 DEFPROCdisplay
 1030 CLS
 1040 PRINT
 1050 FOR I=129 TO 134
 1060  PRINTCHR$(I)"     WEST SOMEWHERE COMPUTER CL
UB"
 1070  NEXT
 1080 PRINT
 1090 PRINTCHR$(129);CHR$(157)
 1100 PRINTCHR$(129);CHR$(157);CHR$(131);CHR$(141)
;"    COMPUTERS       COMPUTERS"
 1110 PRINTCHR$(129);CHR$(157);CHR$(131);CHR$(141)
;"    COMPUTERS       COMPUTERS"
 1120 PRINTCHR$(129);CHR$(157)
 1130 PRINT
 1140 FOR I=129 TO 134
 1150  PRINTCHR$(I)"     WEST SOMEWHERE COMPUTER CL
UB"
 1160  NEXT
 1170 PRINT:PRINT
 1180 PRINT"     ";CHR$(135);CHR$(157);CHR$(129);"
   PRESS THE SPACE BAR     ";CHR$(156)
 1190 G=GET
 1200 IF G<>124 AND G<>32 THEN 1190
 1210 ENDPROC
 1220 REM
 1230 REM Converts all lower case
```

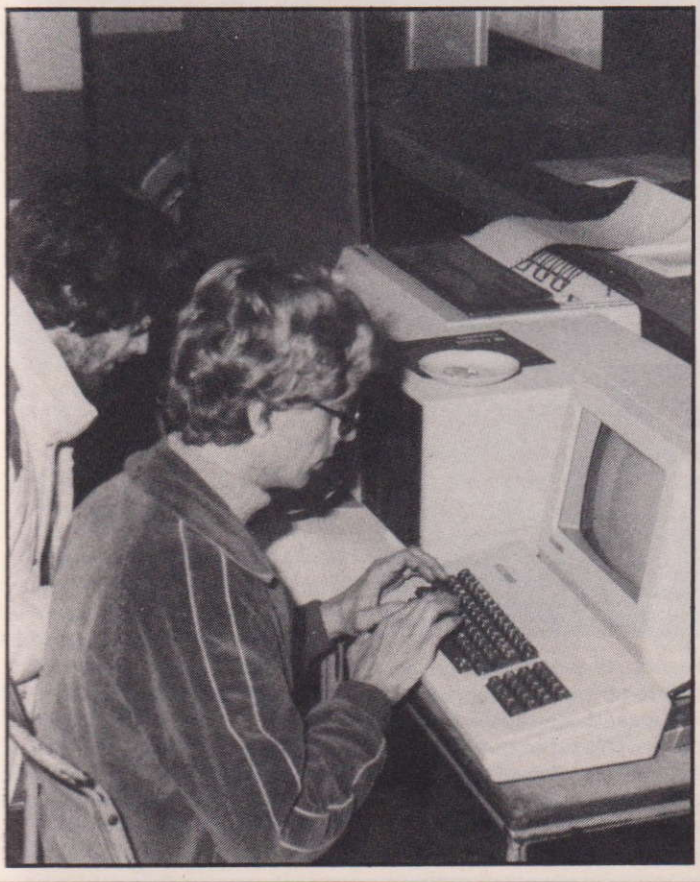
```
 1240 REM inputs to upper case for
 1250 REM later printing uniformity
 1260 REM
 1270 DEFPROCconvtoUC
 1280 LOCAL I,L,T$
 1290 FOR I=1 TO LEN(TEMP$)
 1300  L=ASC(MID$(TEMP$,I,1))
 1310  IF L>96 AND L<123 L=L-32
 1320  T$=T$+CHR$(L)
 1330  NEXT
 1340 TEMP$=T$
 1350 ENDPROC
 1360 REM
 1370 REM Questionaire routine
 1380 REM
 1390 DEFPROCquest
 1400 VDU26
 1410 CLS
 1420 PRINTTAB(0,4);CHR$(134);" To help us provide
 what";CHR$(132);"you";CHR$(134);"want,"
 1430 PRINTCHR$(134);" it will assist if you compl
ete"
 1440 PRINTCHR$(134);" the following short questio
naire."
 1450 PRINT
 1460 PRINT
 1470 PRINTCHR$(134);" We want to know what subjec
ts you"
 1480 PRINTCHR$(134);" would like to see covered m
ost often"
 1490 PRINTCHR$(134);" during future evenings."
 1500 PRINT:PRINT
 1510 PRINTCHR$(134);" Mark the list of items that
 follows"
 1520 PRINTCHR$(134);" on a scale of";CHR$(129);"0
";CHR$(134);"to";CHR$(129);"9"
 1530 PRINTCHR$(129);" 0";CHR$(134);"- no interest
,";CHR$(129);"9";CHR$(134);"- most interest"
 1540 PRINT:PRINT:PRINT:PRINT
 1550 PRINTCHR$(129);"         PRESS SPACE BAR"
 1560 G=GET:IF G<>32 THEN 1560
 1570 CLS
 1580 PRINT
 1590 PRINT
 1600 PRINTCHR$(130);"1.";CHR$(134);" Video entert
ainment"
 1610 PRINT
 1620 PRINTCHR$(130);"2.";CHR$(134);" Computers in
 education"
 1630 PRINT
 1640 PRINTCHR$(130);"3.";CHR$(134);" Business com
puting"
 1650 PRINT
 1660 PRINTCHR$(130);"4.";CHR$(134);" Computer pro
gramming"
 1670 PRINT
 1680 PRINTCHR$(130);"5.";CHR$(134);" Survey befor
e buying"
 1690 PRINT
 1700 PRINTCHR$(130);"6.";CHR$(134);" Electronics
and construction"
 1710 PRINT
 1720 PRINTCHR$(130);"7.";CHR$(134);" Discussion a
nd demonstrations"
 1730 PRINT
 1740 PRINTCHR$(130);"8.";CHR$(134);" Help with sp
ecific problem"
 1750 VDU28,35,24,36,1
 1760 CLS
 1770 FOR I=1 TO 8
 1780 PRINT
 1790  G=GET:IF G<48 OR G>57 THEN 1790
 1800  PRINTCHR$(G)
 1810  inter(I)=G-48
 1820  NEXT
 1830 VDU28,0,24,39,21
 1840 PRINT"     Are the above answers correct"
 1850 PRINT"     If yes press";CHR$(129);"Y";CHR$(1
35);"if no press";CHR$(129);"N"
 1860 G=GET:IF G=78 OR G=110 THEN CLS:GOTO1750
 1870 IF G<>89 AND G<>121 THEN 1860
 1880 FOR I=1 TO 8
 1890  RECORD(count,I)=inter(I)
 1900  NEXT
 1910 ENDPROC
 1920 REM
>
 1930 REM Finds out which type of
 1940 REM micro if any is owned
 1950 REM
 1960 DEFPROCmachine
 1970 VDU26
 1980 CLS
 1990 PRINTTAB(4,4)CHR$(134);"Do you have a comput
er?"
 2000 PRINTTAB(4)CHR$(134);"Type in the appropriat
e number"
 2010 PRINT
 2020 FOR I=0 TO 9
 2030  PRINTCHR$(130);I;CHR$(134);"     ";mach$(I)
 2040  NEXT
 2050  PRINT:PRINTTAB(4)" Enter number "
 2060 G=GET:IF G<48 OR G>57 THEN 1970
 2070 IF G=57:CLS:PRINTTAB(10,10)CHR$141"REALLY!":
PRINTTAB(10,11)CHR$141"REALLY!":PRINTTAB(10,18)CHR
$(129);"BRING IT NEXT WEEK!":FOR I=1 TO 5000:NEXTI
:GOTO 1980
 2080 RECORD(count,0)=G-48
 2090 ENDPROC
 2100 REM
 2110 REM Prints out record for each
 2120 REM individual to keep
 2130 REM
 2140 DEFPROCoutput
 2150 VDU26
 2160 CLS
 2170 PRINTTAB(0,4)
 2180 PRINTCHR$(141);"         THANK YOU"
 2190 PRINTCHR$(141);"         THANK YOU"
 2200 PRINT:PRINT:PRINT
 2210 PRINTCHR$(134);"  YOU MAY LIKE A NOTE OF YOU
R"
 2220 PRINTCHR$(134);"  ENTRIES TO KEEP,"CHR$(129)
;"PLEASE TAKE"
 2230 PRINTCHR$(129);"  A COPY FROM THE PRINTER"
 2240 FOR I=1 TO 2000:NEXT
 2250 PRINT:PRINT:PRINT:PRINT
 2260 VDU2
 2270 VDU1,27,1,66
```

CONTINUED OVER

PROGRAM LISTING

```
 2280 PRINT"     Name"
 2290 VDU1,31
 2300 PRINT"     ";RECORD$(count)
 2310 PRINT
 2320 VDU1,30
 2330 PRINT"     Subject                    Score"
 2340 PRINT
 2350 PRINT"     Video entertainment        ";RECORD(c
ount,1)
 2360 PRINT"     Computers in education     ";RECORD(c
ount,2)
 2370 PRINT"     Business computing         ";RECORD(c
ount,3)
 2380 PRINT"     Programming                ";RECORD(c
ount,4)
 2390 PRINT"     Survey before buying       ";RECORD(c
ount,5)
 2400 PRINT"     Electronics & const.       ";RECORD(c
ount,6)
>
 2410 PRINT"     Discusion & demos.         ";RECOR
ount,7)
 2420 PRINT"     Specific problem           ";RECOR
ount,8)
 2430 PRINT
 2440 PRINT"     Owns _ ";mach$(RECORD(count,0))
 2450 VDU1,10,1,10,1,10,1,10,1,10,1,10
 2460 VDU3
 2470 ENDPROC
 2480 REM
 2490 REM Saves all the data gathered
 2500 REM to cassette file
 2510 REM
 2520 DEFPROCsave
 2530 LOCAL I,J,F
 2540 CLS
 2550 PRINTTAB(6,4)"LOAD CASSETTE"
 2560 PRINTTAB(6)"Press space bar"
 2570 G=GET:IF G<>32 THEN 2570
 2580 CLS
 2590 PRINTTAB(9,8)"SAVING class"
 2600 PRINT:PRINT:PRINT"          ";
 2610 F=OPENOUT("class")
 2620 PRINT£F,count
 2630 FOR I=1 TO count-1
 2640  PRINT£F,RECORD$(I)
 2650  FOR J=0 TO 8
 2660   PRINT£F,RECORD(I,J)
 2670   NEXT
 2680  NEXT
 2690 CLOSE£F
 2700 CLS
 2710 ENDPROC
 2720 REM
 2730 REM Loads a cassette file back
 2740 REM into memory
 2750 REM
 2760 DEFPROCload
 2770 CLS
 2780 PRINTTAB(6,4)"LOAD TAPE"
 2790 PRINTTAB(6)"Press space bar"
 2800 G=GET:IF G<>32 THEN 2800
 2810 CLS
 2820 PRINTTAB(8,9)"LOADING class"
 2830 F=OPENIN("class")
 2840 INPUT£F,count
 2850 FOR I=1 TO count-1
 2860  INPUT£F,RECORD$(I)
 2870   FOR J=0 TO 8
 2880    INPUT£F,RECORD(I,J)
 2890    NEXT
 2900   NEXT
 2910 CLOSE£F
 2920 CLS
 2930 ENDPROC
 2940 REM
 2950 REM Prints out list of all
 2960 REM club members & interests
 2970 REM
 2980 DEFPROCprint
 2990 @%=0
 3000 CLS
 3010 VDU2
 3020 VDU1,27,1,66,1,31
 3030 PRINT"CLUB MEMBERS"
 3040 VDU1,30
 3050 PRINT
 3060 PRINT"No. Name";TAB(20)"Machine   1   2   3
  4   5   6   7   8"
 3070 PRINT
 3080 FOR I=1 TO count-1
 3090  PRINTI;TAB(4)RECORD$(I);TAB(20)"   ";RECORD
(I,0);"      ";
 3100  FOR J=1 TO 8
 3110   PRINTRECORD(I,J);"   ";
 3120   NEXT
 3130  PRINT
 3140  NEXT
 3150 VDU3
 3160 ENDPROC
 3170 REM
 3180 REM Terminates run and resets
 3190 REM cursor ctrl & auto rept
 3200 REM
 3210 DEFPROCend
 3220 CLS
 3230 PRINTTAB(15,12)"PRESS SPACE BAR TO END"
 3240 G=GET
 3250 IF G<>32 ENDPROC
 3260 *FX4,0
 3270 *FX12,0
 3280 CLS:PRINTTAB(12,15)"END":END
 3290 REM
 3300 REM Calculates %age and plots
 3310 REM piechart
 3320 REM
 3330 DEFPROCpiechart
 3340 VDU19,0,6,0,0,0,19,1,5,0,0,0,19,2,3,0,0,0,19
,3,4,0,0,0
 3350 VDU5
 3360 GCOL0,128:CLG
 3370 R=400:x=650:y=460:inc=.1:Z=0:col=2
 3380 FOR J=1 TO 8
 3390  FOR I=1 TO count-1
 3400   sect(J)=sect(J)+RECORD(I,J)
 3410   NEXT
```

```
 3420  sect(J)=sect(J)/(count-1)
 3430  IF sect(J)=0 sect(J)=.1
 3440  NEXT
 3450 tot=0
 3460 FOR I=1 TO 8
 3470  tot=tot+sect(I)
 3480  NEXT
 3490 ang=360/tot
 3500 FOR I=1 TO 8
 3510  sect(I)=sect(I)*ang
 3520  NEXT
 3530 seg=0
 3540 FOR I=1 TO 8
 3550  p=0
 3560  prin=seg+RAD(sect(I)/2)
 3570  seg=seg+RAD(sect(I))
 3580  MOVEx,y
 3590  col=col+1:IF col=4 col=2
 3600  GCOL0,col
 3610  REPEAT
 3620   MOVE R*SINZ+x,R*COSZ+y
 3630   Z=Z+inc
 3640   PLOT 85,R*SINZ+x,R*COSZ+y
 3650   IF Z>=prin AND p=0:GCOL0,1:MOVE(R+45)*SINZ
+(x-10),(R+45)*COSZ+y+10:PRINTI:GCOL0,col:p=1
 3660   MOVEx,y
 3670   UNTILZ>=seg
 3680  NEXT
 3690 MOVE200,1000:GCOL0,1:PRINT"YOUR INTERESTS"
```



```
 3700 G=GET:IF G<>32 THEN 3700
 3710 ENDPROC
 3720 REM
 3730 REM Calculates %ages, draws
 3740 REM and labels barchart
 3750 REM
 3760 DEFPROCbarchart
 3770 VDU5
 3780 VDU19,0,6,0,0,0,19,1,0,0,0,0,19,2,1,0,0,0,19
,3,4,0,0,0
 3790 GCOL0,128:CLG
 3800 FOR I=1 TO 8
 3810  sect(I)=0
 3820  FOR J=1 TO count-1
 3830   sect(I)=sect(I)+RECORD(J,I)
 3840   NEXT
 3850  NEXT
 3860 total=(count-1)*9
 3870 FOR I=1 TO 8
 3880  sect(I)=sect(I)/total*100
 3890  NEXT
 3900 GCOL0,1
 3910 X=20:Y=150
 3920 MOVEX+250,Y:DRAW1150,Y
 3930 FOR I=0 TO 10
 3940  MOVEX,Y+5
 3950  GCOL0,2
 3960  IF I<10 PRINT" ";I*10 ELSE PRINTI*10
 3970  IF I=10 GOTO4020
 3980  MOVEX+250,Y
 3990  Y=Y+75
 4000  GCOL0,1
 4010  DRAWX+250,Y:DRAWX+240,Y
 4020  NEXT
 4030  GCOL0,2
 4040 MOVEX+100,Y+50:PRINT"%"
 4050 X=300:Y=100
 4060 FOR I=1 TO 8
 4070  MOVEX,Y
 4080  PRINTI
 4090  X=X+110
 4100  NEXT
 4110 X=276:Y=153:g=2
 4120 FOR I=1 TO 8
 4130  g=g+1:IF g=4 g=2
 4140  GCOL0,g
 4150  PROCbox
 4160  X=X+110
 4170  NEXT
 4180 MOVE 350,990:PRINT"YOUR INTERESTS"
 4190 G=GET:IF G<>32 THEN 4190
 4200 ENDPROC
 4210 REM
 4220 REM Plots each bar of chart
 4230 REM
 4240 DEFPROCbox
 4250 H=sect(I)*7.5
 4260 MOVEX,Y
 4270 MOVEX,Y+H
 4280 PLOT85,X+100,Y+H
 4290 MOVEX+100,Y
 4300 PLOT85,X,Y
 4310 ENDPROC
 4320 DATA None,TRS 80/Video Genie,Pet/Vic 20,Shar
p,Apple,Spectrum/ZX 81,Atari,Acorn/BBC,Superboard/
UK 101,Cray 1
```

Out with Auntie

Making a television programme is not as easy as it sounds. We take a 'behind the scenes' look at the BBC's computer programme series.

We all take for granted the technology and expertise that surrounds us every day—the radio, the hi-fi, the telephone, and most of all the good old 'box'.

But spending a day in a BBC studio made me realise what a complex process making a television show is. I was there to watch an episode of the BBC Computer Programme.

MAKING AN ENTRANCE

After walking down apparently endless corridors in Television House in Shepherds Bush, I was shown into a darkened room that looked like something out of Mission Control at NASA! Six or seven people seated at a desk in front of 33 TV monitors (each showing a different picture of the studio, and some with current BBC and ITV transmissions), large consoles of brightly coloured switches, microphones, voices coming from thin air (or so it seemed) and subdued lighting all added to my now overtaxed sense of awe! This was the Producer's Gallery, the nerve centre of studio operations.

From here the director, Patrick Titley for this programme, directs the entire pro ceedings. He occasionally goes down to the studio floor himself, but normally talks through the microphones directly to the floor manager, who relays all directions to the presenters, camera crew, designer, etc. Because you can't easily see the studio from the gallery, the director gets a true idea of what's happening on-screen.

Photograph courtesy of Robin Mudge

IN PREPARATION

Before anyone gets into a studio, months of hard planning, research, chasing-up and re-planning had to be done by the team. Work on this series began way back in the summer of 1982 when the team started meeting to decide the overall aims and layout of the series before getting down to the nitty-gritty of the specific content of each programme. This in itself must have been a headache.

Series 1 had gone well but as we all know, repeats (or re-hashes) of past programmes or series, no matter how good, do not make for a satisfied viewing public. This series had to be new, informative, appealing to both computer novices and the more experienced, and still be entertaining.

Once the content of a programme has been agreed, —in this case advanced graphics—the team had to decide exactly what points needed to be made (and how) and what type of equipment and examples would be necessary for that purpose. In this particular episode, this was done by contacting manufacturers (and looking at and trying out possible products to see if they would fit the bill). Film clips and on-location filming are also vital ingredients.

ON THE DAY

Although studio work is essentially a team effort, it is obvious that the director rules the roost. Even so, he is always open to suggestion, either from the gallery or the studio floor, but he is the man who firmly guides the operation through.

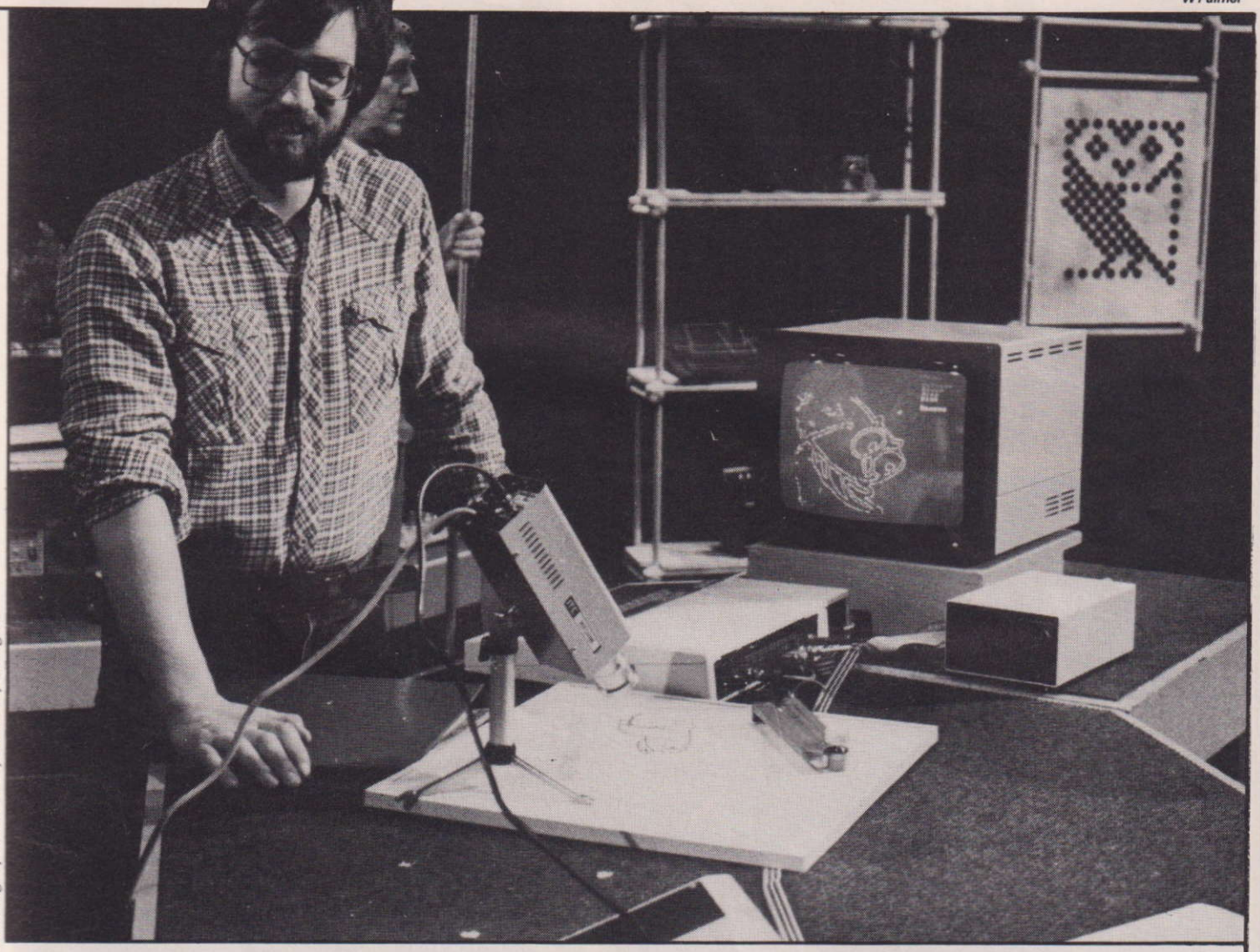
I have not yet mentioned the producer, David Allen. That's because on the studio day, the director is in charge while the producer takes a back-seat and generally oversees the proceedings, making a few helpful comments now and then and advising where necessary. The producer also spends time discussing details for the next programme, in this case with John Coll, from Acorn Computers who as the BBC Micro manufacturer has quite naturally been involved with the whole series.

WE'LL JUST DO THAT AGAIN

Ian McNaught-Davis and Ian Trackman are the presenters for Programme 9, but they are not merely figure-heads with familiar faces. They have to know what they're talking about and a large part of the software that is used on the set has been written by Ian Trackman.

The script is rehearsed and recorded in blocks, but I use the word 'script' rather loosely.

W Palmer



Photographs courtesy of Henry Budgett

Everyone has a copy of a script that has been written and re-written countless times, but very little of the programme is pre-written on the autocue (the prompt boards used by presenters). The script may just say "Mac describes the set-up" or "Ian explains that there is a mathematical model of the house stored in the computer's memory" and it is up to the presenter to put into his or her own words. This gives an informal and conversational atmosphere to the programmme giving rise to amusing, if not always 'air-worthy' quips during rehearsal!

Rehearsal and recording are fairly stop/start matters for all sorts of reasons. For instance, a piece of software won't LOAD properly, the director wants a repeat of a section so that he can check the camera shots, the presenter wants to say the same thing but explain it in just a slightly different way . . . And those are just a few.

This is also the time to make sure that the equipment is to hand when needed and that the overall effect is working. Things like: Would it be better if I stood closer to the table? Which is the best way for me to point at the computer screen?, etc, as well as the chance to run the pre-recorded film clips and check that all the credits are right.

TIMING IT RIGHT

The atmosphere during the whole day seems informal and relaxed which hides the ever-present pressure involved.

Quietly working away are the production assistants who carefully time every piece individually and the whole thing generally, since time is absolutely of the essence. The actual sequences that are recorded as such have to be strictly timed, since the programme has to a specific length within extremely fine limits, but also studio time itself is very precious. When it comes time to recording a sequence things become a little more serious, with the technical manager, Tony Bate, getting the video people ready to record. This he does via the phone connecting him to other groups of technicians in other parts of the building and when they're ready to go you hear those immortal words "Get ready, Opening sequence, cue Mac and Action!"

CONTINUED OVER

The presenters then 'present' the sequence while in the gallery the director is scanning the TV monitors to see not only what is actually being recorded but also what can be seen with the other three cameras. He then calls the time for a change of camera at a particular instant to the vision mixer, Susan Brincat, who operates the required levers and buttons.

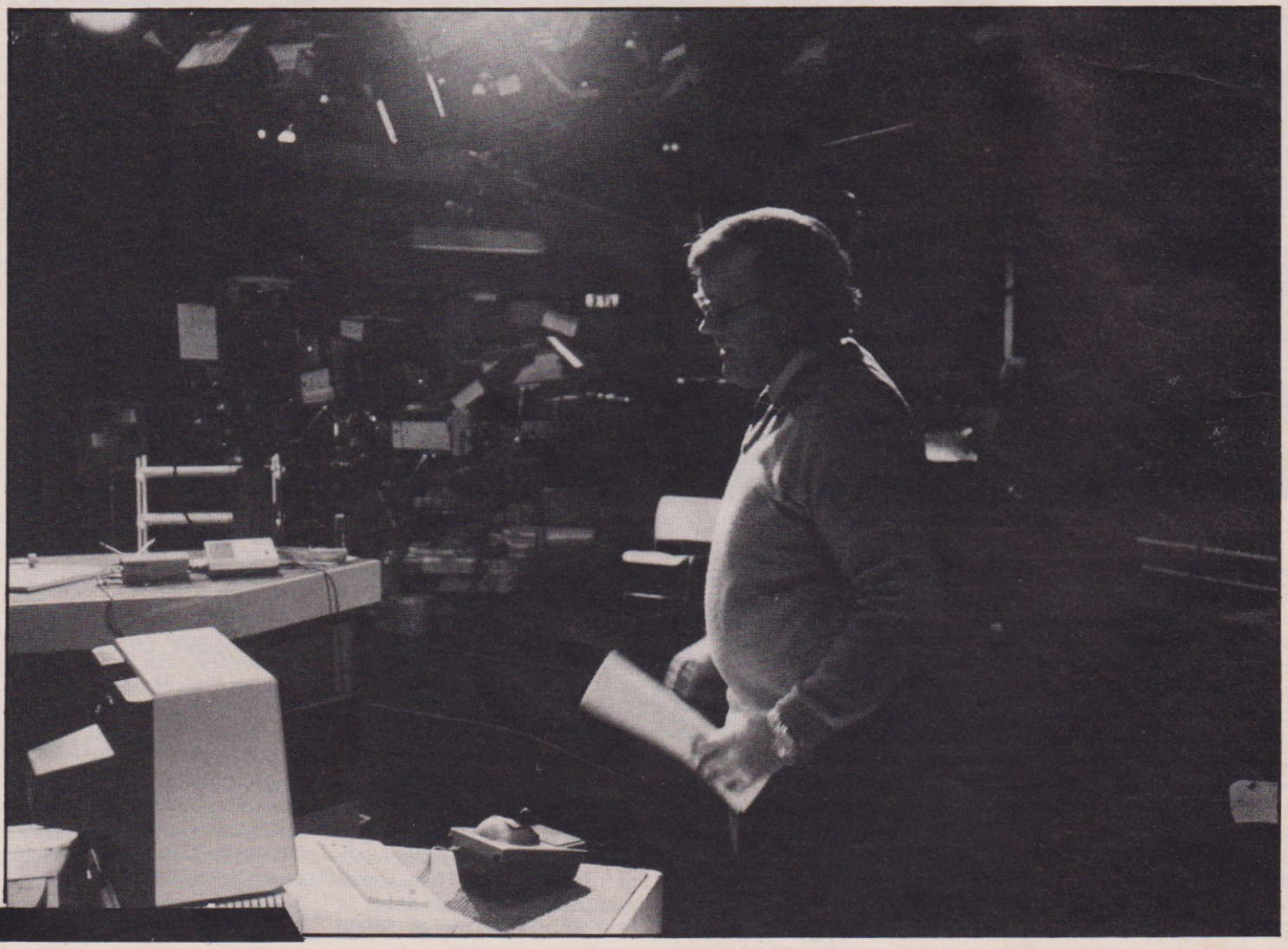
A 'take' may go perfectly with respect to the presentation and technical execution, but it is not usual to make do with just one 'take'. The director will often request a sequence or part of a sequence to be re-shot, so that he will have plenty of scope when it comes to editing the pro- next day. Eventually, by repeatedly rehearsing and recording, the whole script is worked through and when the director feels that he has enough on tape to form the whole programme, he thanks everyone and there it is—'in the can'. Even I felt a great sense of achievement when it was finished—and I hadn't been involved at all!

EPILOGUE

The day that I spent in the studio went fairly smoothly, although I was assured that this is not always the case. Sometimes it must be like the TV series 'It'll Be All Right On The Night'. For instance, there was the time when a pieceof equipment suddenly developed a mind of its own and an apparently smooth sequence in the programme had been made up from inumerable shots taken when the equipment was being not quite so temperamental. That's why good editing is so important. I also had to bear in mind that the eight hours that I had spent observing in the studio and all those previous months of hard work and planning had resulted in about 26 minutes' worth of television air time!

My day in the studio was quite a revelation and now watching TV will never be the same. I hope that I never take TV quite so much for granted in the future.

My thanks go to the whole team present on the day, including those I have not specifically mentioned (for no reason other than lack of space), not only for letting me take up some of their valuable time and space in the gallery, but also for making it a very interesting and enjoyable day.



Photograph courtesy of Henry Budgett

THE SOUND OF MUSIC

B Landsberg

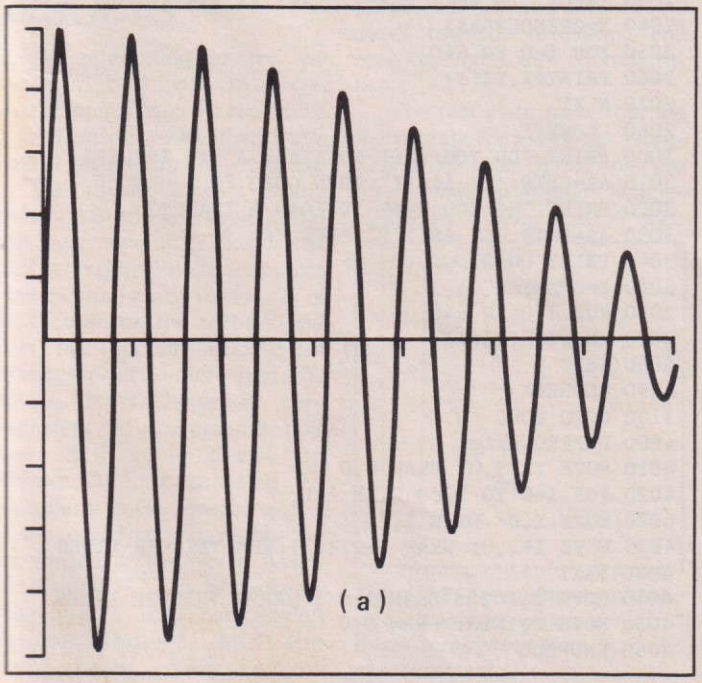
Follow our advice and turn your micro into a realistic music machine.

Sound is an incredibly complex phenomenon, and even simple one-note instruments produce very complicated wave shapes. Every note consists of many harmonics (other notes sounding at frequencies which are often exact multiples of the note itself) each with its own variations of volume and pitch with time. Sometimes over 20 of these harmonics may contribute significantly towards giving the note its particular tone quality. It may then be easily understood why the BBC Micro, even with its SOUND and rather elaborate ENVELOPE commands, cannot hope to imitate musical instruments in a truly realistic manner. It is the aim of this article to present a simple circuit which converts a sound into its amplitude envelope and uses the BBC Micro (Model B only, I'm afraid) to capture and store it in order to assist in the production of more realistic ENVELOPE commands.

The ENVELOPE command on the BBC Micro may be used to control the way any note varies its amplitude and pitch with time, but it is amplitude variations only that will be discussed here. In order to demonstrate what is meant by the term 'amplitude envelope' and how it may be measured, consider Fig. 1a which shows a typical wave sound varying in amplitude. Note that the signal goes both positive and negative, and the average is approximately zero. The first step in treating a wave like this is called rectification, and consists of turning each negative signal into an equivalent positive signal—if we could do this in BASIC we would merely use the expression, V = ABS(V)!

The rectified signal is shown in Fig. 1b, and must be smoothed so that only the amplitude envelope as shown in Fig. 1c remains. If you are interested only in the results, please skip the next two sections.

Fig. 1a. A typical sound wave of varying amplitude.



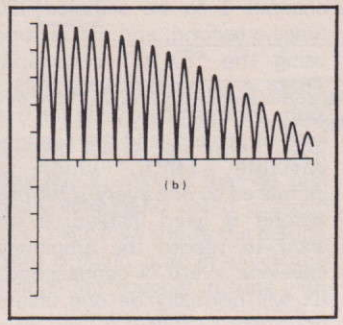


Fig. 1b. The signal after the rectification process.

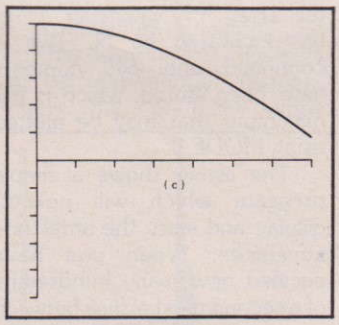


Fig. 1c. The amplitude envelope of the sound wave after rectification and smoothing.

ELECTRONIC CIRCUITRY

Most BBC Micro users will own a cassette recorder, and many of these are fitted with a built-in microphone. I use a HITACHI TRQ-299, and a signal with a peak-to-peak voltage of about 0.8 Volts is produced at the 'EXT. SP.' socket when it is in record mode and when a reasonably loud sound is directed at the microphone. The electronics were designed around this kind of output, but may easily be altered if your microphone gives a different response. It ought to be mentioned at this stage that a microphone is not even needed for this project as long as the sounds to be studied have been pre-recorded onto cassette—the signal may be taken out of the exterior speaker socket when the cassette is played back.

The simplest way to rectify an AC signal is to use four diodes connected together to form a rectifier bridge, but I have not done this because the first 0.2 - 0.6V (depending on which type of diodes are used) is lost using this method. Instead, the circuit shown in Fig. 2, which does not lose any of the signal, was constructed.

For the power lines, you can simply use two batteries of 3V or more, and the resistor R1 and capacitor C1 are needed to ensure that there is no DC bias on the signal. If you are absolutely certain that the output from your microphone has no DC bias, these two components may be omitted. The bulk of the rest of the circuit performs the rectification, and the resistor R7 and capacitor C3 smooth the signal. Finally, the diode D3 is there to protect the BBC Micro from any negative voltage input should something go wrong. The ground and positive output leads should be connected to pins 8 and 15 respectively of the analogue input, which represent channel 1 of the analogue to digital conversion input.

A compromise between smoothing of the waveform and response time of the envelope must be made—obviously, we cannot fully smooth 100Hz and expect the envelope to respond to events lasting 1/100 of a second! The values shown for R7 and C3 give rise to an unwanted ripple of 1% at 400Hz, but with these values the system takes about 1/30 of a second to fully

CONTINUED OVER

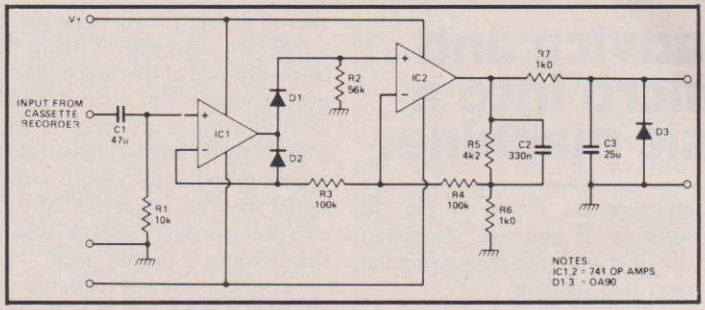


Fig. 2. The electronic circuit which converts the signal from the microphone to its amplitude envelope. The output lines go to pins 8 and 15 of the analogue input socket of the BBC Micro.

respond to a change in the amplitude of the signal. This seems a good enough compromise, but if you wish to follow even faster processes using this method, the value of R7 or C3 should be reduced—but this will increase the ripple and may add a small random effect to measurements at lower freqencies. There are more complex circuits by which this problem may be overcome, but the circuit shown is adequate for examining normal amplitude envelopes.

Finally, the values of the resistors R5 and R6 determine the amplification of the amplitude envelope. The BBC Micro will accept analogue input in the range 0-1.8V, and if the rectified signal is just under 0.4V, a gain of five should be more than adequate. The value of R5 should be increased for greater amplification and decreased for lower.

INTERFACING SOFTWARE

Having connected the interfacing system, we need a program to collect the data, display it on the screen and, if desired, store it on cassette. The four ADC channels are sampled in turn every 1/100 of a second, but disabling channels 2-4 allows channel 1 to be sampled 100 times a second, and this is done using the *FX16,1 command. There is no need to write timing loops in assembler code for this kind of project, and the pseudo-variable, TIME, which is increased by one every 1/100 of a second is used instead. If we want to record the amplitude envelope every N centiseconds (N will normally be one unless you want to follow a very slow motion over a long time) we must store the data from channel 1 in the array Y% whenever TIME has increased by N. This is continued until 640 numbers have been stored, which is the maximum that may be plotted using MODE 0.

The listing shows a simple program which will collect, display and store the amplitude envelopes. When you have decided how many hundredths of a second must elapse between samples, pressing the Return key will initiate the whole procedure which will, in effect, allow the BBC Micro to act as a low-frequency storage oscilloscope. After 6.4 seconds (per 1/100 of a second between samples) the screen will clear and a tracing of the envelope appears on the screen. If for some reason it was too loud or too soft (or your dog barks at the wrong time), a new trace may be started. When a trace worthy of keeping eventually appears, it may be saved as a file on cassette—for those with working discs, I envy you, but it is trivial to modify the program to do this. Finally, a plot of the logarithm of the amplitude

PROGRAM LISTING

```
 10 REM **    CONTROL PROGRAM FOR INTERFACE      **
 20 REM **  ALSO DISPLAYS, SAVES OR LOADS TRACES **
 30 *FX16,1
 40 DIMY%(640): FORI=1TO640: Y%(I)=0: NEXT: MODE 0
 50 PRINT"DO YOU WANT TO START A NEW TRACE?"
 60 A$=GET$: IF A$<>"Y" THEN GOTO 3020
 70 CLS: PRINT"HOW MANY CENTISECONDS PER STEP?"
 80 INPUT N: IF N<1 GOTO 70
 90 TIME=0
100 X=ADVAL(1): I=TIME/N
110 IF I>640 THEN GOTO 1000 ELSE Y%(I)=X
120 GOTO 100
1000 PRINT CHR$(7);CHR$(7);CHR$(7)
1010 MODE 0: PROCaxis
1020 FORI=0TO640
1030 DRAW I*2,Y%(I)/64
1040 NEXT
1050 A$=GET$
1060 PRINT "DO YOU WANT TO SEE A LOG PLOT OF THIS?"
1070 A$=GET$
1080 IF A$<>"Y" THEN GOTO 2010
1090 MODE 0: PROCaxis
1100 SCALE = 1024/LOG(65536)
1110 FOR I=0 TO 640
1120 IF Y%(I)>0 THEN Y=LOG(Y%(I)) ELSE Y=0
1130 DRAW I*2,Y*SCALE
1140 NEXT
2000 A$=GET$
2010 PRINT"DO YOU WANT TO SAVE THIS FILE?"
2020 A$=GET$: IF A$<>"Y" GOTO 3000
2030 INPUT "TO WHAT NAME? " A$: IF A$="" GOTO 2030
2040 X=OPENOUT(A$)
2050 FOR I=0 TO 640
2060 PRINT#X,Y%(I)
2070 NEXT
2080 CLOSE#X
3000 PRINT "DO YOU WANT TO START A NEW TRACE?"
3010 A$=GET$: IF A$="Y" THEN GOTO 70
3020 PRINT "DO YOU WANT TO LOAD A TRACE?"
3030 A$=GET$: IF A$<>"Y" THEN END
3040 PRINT"LOADING..."
3050 X=OPENIN("")
3060 FOR I=0 TO 640
3070 INPUT#X,Y%(I)
3080 NEXT
3090 CLOSE#X
3100 GOTO 1000
4000 DEFPROCaxis
4010 MOVE 1279,0: DRAW 0,0
4020 FOR I=0 TO 1280 STEP 100
4030 MOVE I,0: DRAW I,10
4035 MOVE I+2,0: DRAW I+2,10 : REM THICKER LINES
4040 NEXT
4045 MOVE 2,1023: DRAW 2,0 : REM   THICKER LINES
4050 MOVE 0,1023: DRAW 0,0
4060 ENDPROC
```

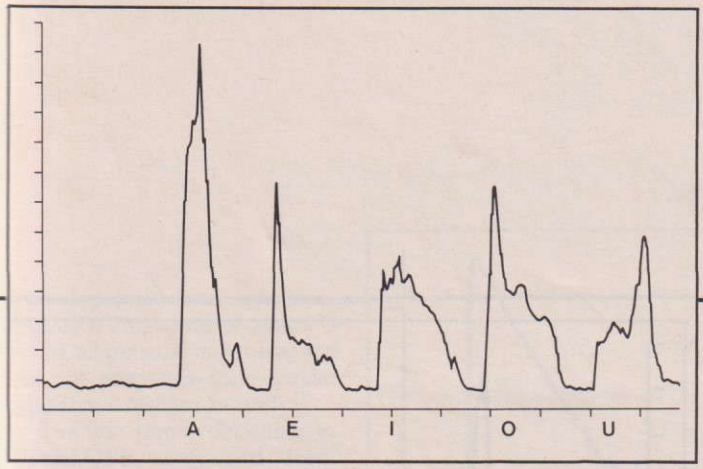


Fig. 3a. The amplitude envelope of the five vowels.

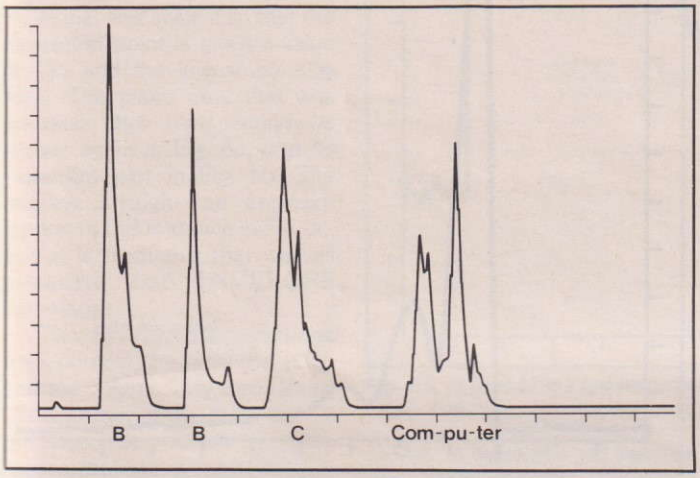


Fig. 3b. The amplitude envelope of the words 'BBC Computer' spoken very slowly.

may be displayed—the significance of this will be discussed later.

RESULTS

Before attempting to try the system out on musical instruments, I used my own voice to familiarise myself with it. Some typical results are shown in Fig. 3. The upper diagram is the envelope of the five vowels, and the lower one is of the words 'BBC computer'. It is evident that I put more stress on the first 'B' of 'BBC' than on the second, and also the second syllable of 'computer' is louder than the others. It is, however, very important to understand that even if I could devise an ENVELOPE command to reproduce these graphics, the result could not possibly be recognised as the words which were spoken. This is because speech is a very complicated subject indeed, and the frequences used are of paramount importance in our recognition of what is being said. For musical instruments, however, some practical progress has been made in some cases towards measuring and reproducing their characteristics.

The first instrument I turned my attention to was the piano. This article was, in fact, inspired by certain previous articles which have appeared in well-known magazines and which describe envelopes that no self-respecting piano could possibly give. Figure 4a shows the results from a piano note in which the strings are roughly in tune with each other (as most notes on the piano consist of two or three strings being struck simultaneously). The note rises quickly to its maximum amplitude and then dies away in what physicists call an exponential fashion. Actually, there are a few humps and bumps towards the end of the curve as always happens in real experiments of this nature, but by and large the last sentence is true. Figure 4b shows the result of playing the same note, but releasing the piano key after about two seconds. It can be seen that the amplitude envelope is not simple, but the note takes a further half second to die away. All of this information is useful in constructing more accurate ENVELOPE commands.

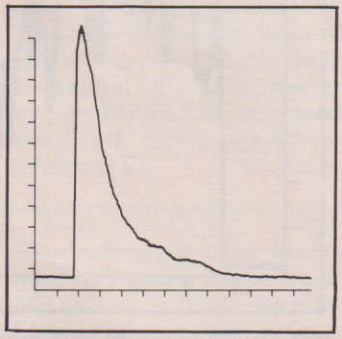


Fig. 4a. The amplitude envelope of a piano note which is held down throughout the measurement.

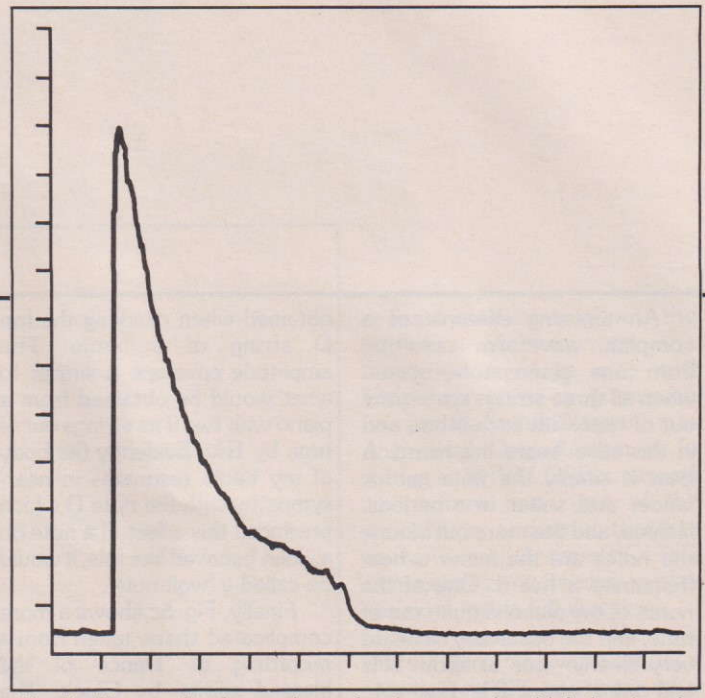


Fig. 4b. The amplitude envelope of a piano note in which the piano key is released after about two seconds.

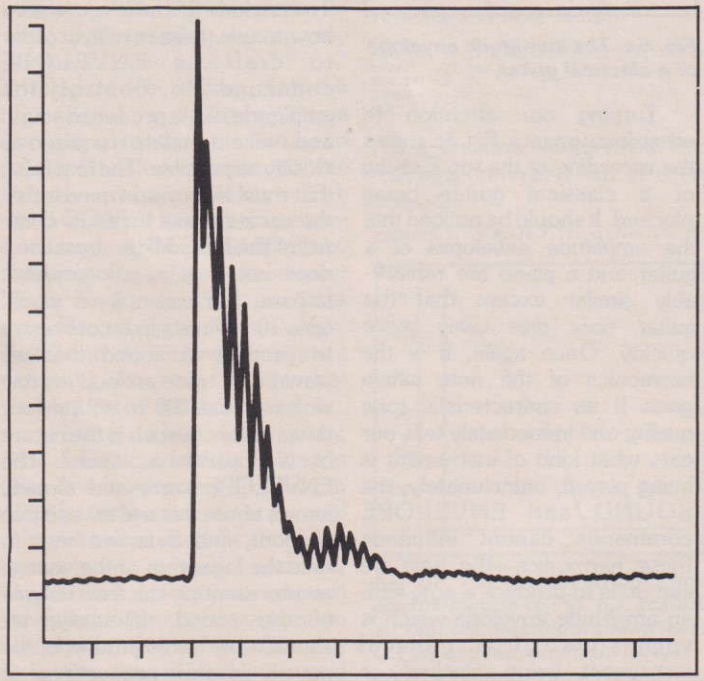


Fig. 4c. The amplitude of a piano note in which the strings are slightly out of tune with each other.

CONTINUED OVER

An amusing example of a complex waveform resulting from one piano note occurs when all three strings are slightly out of tune with each other, and in this case 'beats' are heard. A beat is simply the note getting louder and softer in a periodic fashion, and the more out of tune the notes are the faster a beat frequency is heard. One of the notes of my piano is quite out of tune, and the beats may be heard very clearly—one at about 8Hz and one at about 1Hz. Figure 4c shows the measured amplitude envelope—if your piano gives a result like this then get it tuned immediately!

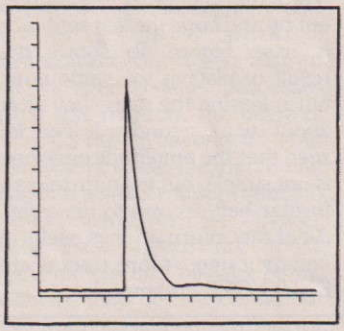


Fig. 5a. The amplitude envelope of a classical guitar.

Turning our attention to other instruments, Fig. 5a shows the recording of the top E string of a classical guitar being plucked. It should be noticed that the amplitude envelopes of a guitar and a piano are remarkably similar except that the guitar note dies away more quickly. Once again, it is the harmonics of the note which gives it its characteristic tone quality and immediately tells our ears what kind of instrument is being played; unfortunately, the SOUND and ENVELOPE commands cannot influence these harmonics—the best we can do is to produce a note with an amplitude envelope which is suggestive of a musical instrument being plucked or struck.

Figure 5b shows the results obtained when plucking the top D string of a banjo. The amplitude envelope is similar to what would be obtained from a piano with two if its strings out of tune by 1Hz. Evidently the body of my banjo resonates in near-sympathy with the note D which produced this effect. If a note on a violin behaved like this, it would be called a 'wolf-note'.

Finally, Fig. 5c shows a more complicated shape taken from a recording of 'Dance of the blessed spirits' by Gluck. The two braces mark two flute notes which are played as a solo before the orchestra comes in again. It should be noticed that there is an amplitude modulation of about 5Hz on the note of the flute, and although this may be accompanied by a pitch modulation too, our apparatus as it stands cannot detect this!

NEW ENVELOPES FOR OLD

To conclude this article, consider how to use these results in order to draft an ENVELOPE command to control the amplitude of the produced sound and make it imitate my piano as closely as possible. The first thing that must be borne in mind is that the ear is not like the ADC channel of the BBC Micro, because it does not hear in a logarithmic fashion. For example, it would take 10 violinists in an orchestra to produce a sound that we would call 'twice as loud' as one violinist, and 100 to sound four times as loud—such is the nature of a logarithmic scale! The ENVELOPE command already knows about this and takes it into account, and thus, we have to take the logarithm of the voltage as representing the true volume of the sound. (Actually, we should take the logarithm of the power, which is proportional to the square of the voltage, but the answer will come out the same

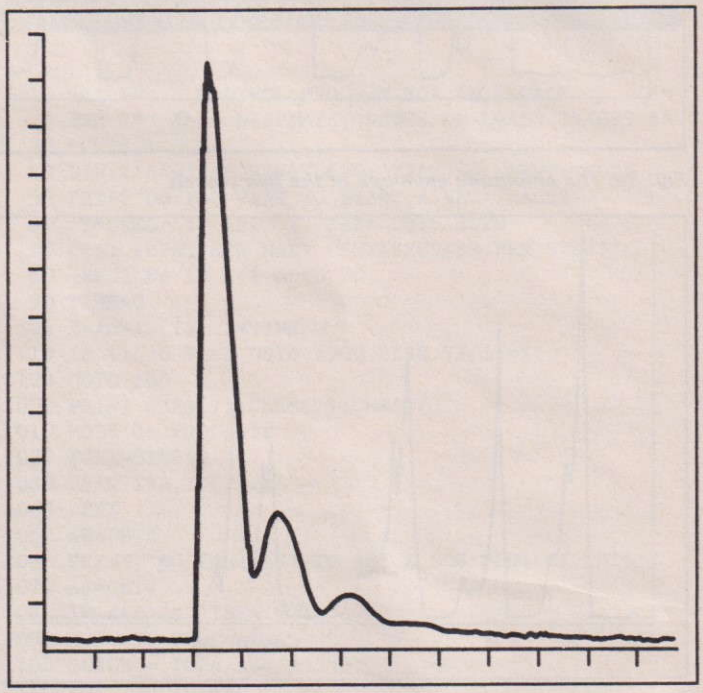


Fig. 5b. The amplitude envelope of a banjo.

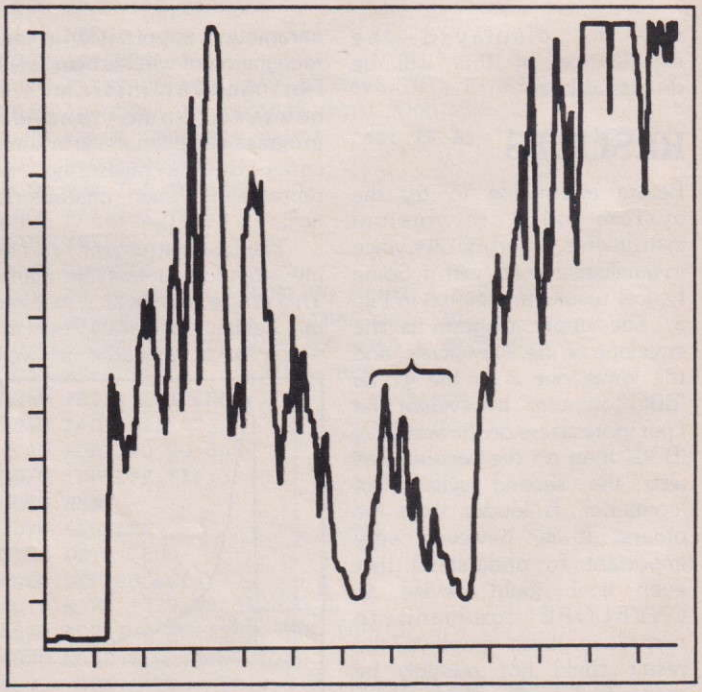


Fig. 5c. The amplitude envelope of a recording of 'Dance of the Blessed Spirits' by Gluck. The flat portions at the top of the screen are where the input voltage rose to a level higher than the BBC Micro could process (about 1.8V). The braces mark two flute notes from which we may gain information on their amplitude envelopes.

anyway!) This state of affairs is not all all unusual in nature, and the eye responds in a similar logarithmic fashion as well.

The first step in designing an ENVELOPE command from these results is thus to plot the logarithm of the amplitude envelope, and scale it so that the maximum point is given a value of 126 and the lowest point is zero. The piano note that was released after two seconds is shown again in Fig. 6a, and its logarithm plot in Fig. 6b. The nearest straight-line approximation to this is shown in Fig. 6c, and it is from this that we will generate our ENVELOPE commands.

The ENVELOPE command may control the way the pitch and the volume vary throughout the duration of the note, and to program the volume we may regard the note as consisting of four separate stages called attack, decay, sustain and release. Briefly, the attack phase determines how quickly the note rises to its maximum amplitude, and this is followed by the decay phase in which its volume decreases until it reaches a target level. The sustain phase, which lasts from this point until the end of the note, is where the volume remains at the same level (or decreases at a different rate to the decay phase). When the note has reached the end of its time, it is then in its release phase and

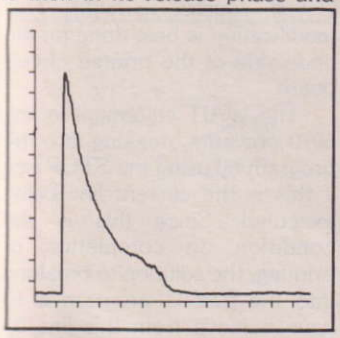


Fig. 6a. The amplitude envelope of a piano note in which the piano key is released after about two seconds.

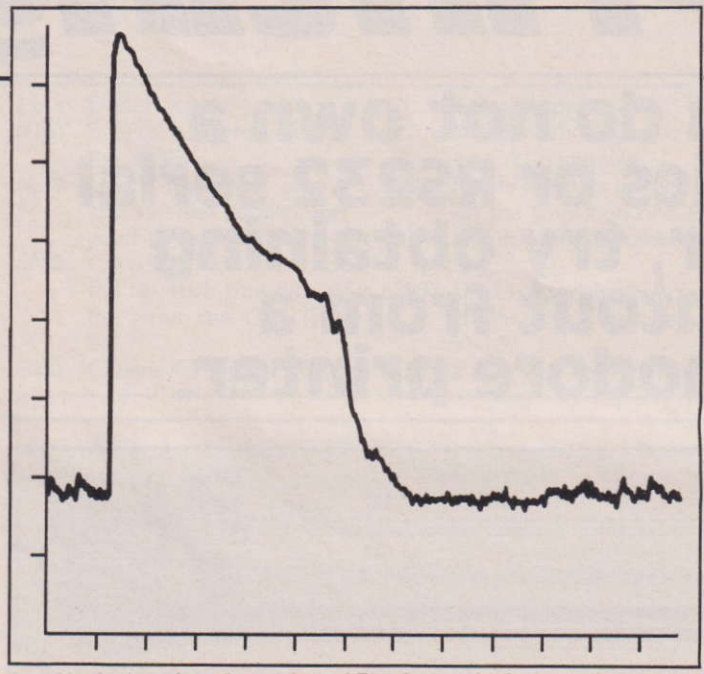


Fig. 6b. A plot of the logarithm of Fig. 6a in which the scale has been expanded to accommodate only the upper portion of the trace.

decays to silence at yet another rate. For full details, the BBC's **User Guide** provides an excellent coverage of this topic.

If we now look at Fig. 6c, we see that the piano note has a fast attack which takes about three centiseconds followed by a decay to zero which would have taken around three and a half seconds if I had not released the piano key. As this represents a fall from a volume level of 126 to zero, the rate of decay is about one step every three centiseconds. Our measurements of the piano note show that it has no sustain phase at all! Finally, when the piano key has been released, the volume of the note decays roughly three times as fast. Without any further ado, an ENVELOPE command which will satisfy the above conditions is:

```
ENVELOPE 1,3,0,0,0,0,0,0,126,-1,0,-3,126,0
```

In order to hear its effect on a low note, try typing in the command, SOUND 1, 1, 0, 20, after the ENVELOPE command has been executed. Note that the volume can be heard to decrease in discreet steps; this is because the BBC Micro may sound at only one of 16 volume levels (0 - 15) even though the ENVELOPE command may be programmed as if it had 127. This means that the steps are eight times bigger than they would be under a full implementation of the system, and each step may be heard quite easily.

Figure 5a shows that the guitar gives a similar amplitude envelope to that of the piano except that it decays about three times more quickly. The ENVELOPE command shown above may be modified by altering the second parameter from three to one. This produces a note having a quality which, although attempting to imitate a guitar, is also somewhat suggestive of a harpsichord.

Finally, we attempt to imitate the dynamics of a flute. A look at the not very pretty diagram in Fig. 5c shows that the volume of the note rises and falls about five times per second, and the logarithmic plot (not shown) shows the variation in volume to be about 10–20%. Here we reach our first stumbling-block, and that is although the pitch envelope may be programmed to repeat itself throughout the duration of the note, the amplitude envelope may not! How then can we program such an amplitude modulation into any note? One solution is to design an ENVELOPE command to perform one cycle of the modulation, and string together as many carefully-timed SOUND commands as is needed to play the note for its full duration.

The approach to this is to start at a volume level of 106, let it increase to 126 in a tenth of a second and let it decrease to 106 again, also in a tenth of a second. As the ENVELOPE command starts the note at zero volume, we need to precede it with a short SOUND command in order to set the volume level at around 106 to begin with, and a volume parameter of -13 will do this. The following sequence of instructions will produce a note lasting for about three seconds which, although not sounding quite like a flute, will contain a 5Hz volume modulation of 20% and will decay in a pleasing manner after the time for the note is up:

```
ENVELOPE 1,1,0,0,0,0,0,0,2,-2,0,-4,126,106
SOUND 1,-12,100,1
FOR I=1 to 15
SOUND 1,1,100,4
NEXT I
```

For a shallower modulation, the following command may be used instead:

```
ENVELOPE 1,1,0,0,0,0,0,0,1,-1,0,-4,126,116
```

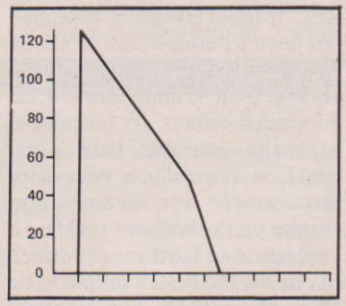
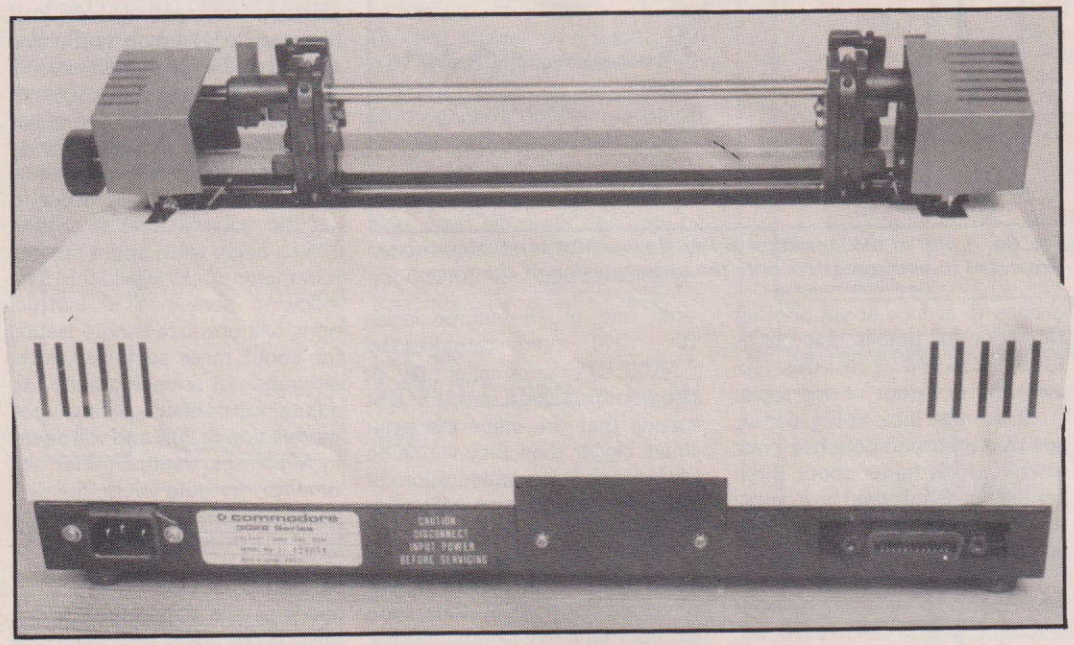


Fig. 6c. The nearest straight-line approximation to Fig. 6b, scaled so that the maximum sound level is 126 and the minimum level is given a value of zero. We may tailor our ENVELOPE commands directly from this plot.

Pet Printing

If you do not own a Centronics or RS232 serial printer, try obtaining printout from a Commodore printer.



This project is suggested as a temporary solution to the problem of hard copy for the BBC (Model B) users who may not have a parallel Centronics or RS232 serial printer, but have access to a Commodore IEEE interfaced printer. Its purpose is to make available hard copy which is normally a necessary requirement for examination course work. Without building a sophisticated hardware interface to change the BBC's output to be PET compatible, two possible solutions were considered.

TWO SOLUTIONS

Either create software to load into the BBC which could carry out the correct handshaking and coding requirements (the PET printer does not use standard ASCII), ie write a printer output routine under *FX5, 3. Or write software to enable the PET to act as an effective, if rather expensive, interface to once again carry out handshaking and coding.

This project uses the latter option because of the relative simplicity and flexibility, but does, of course, mean that a PET must be dedicated to the BBC whilst any printing is required. When printing from the BBC is needed a short program is loaded into the PET which is linked to the printer and a cable is connected between the PET and BBC.

HARDWARE

In the majority of cases it will be fo a single cable connection between the PET and the BBC is all that is required. Entry to the BBC is through the printer plug (PL9) and the PET by way of the user port (J2). The BBC socket required is a 26-way SPEED-BLOC mounting socket (RS467 - 295) for which an assembly tool is recommended (RS468 - 197) and (RS468 - 232). The ribbon cable (RS357 - 873) should be of suitable length to reach between the machines as they stand on a work surface but should, however, be kept as short as possible.

The PET edge connector is a 24-way dual read out type and is available from Commodore dealers. The connections to be made are shown in Fig. 1. The twelve lines from the BBC which are used are 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19 and 20 and they are listed opposite the lower edge connector tabs A - M to which each must be soldered.

SOFTWARE

The program to be loaded into the PET is shown in Listing 1. together with an explanation of the function of each line. This follows the algorithm shown in Fig. 2.

Both upper and lower case printer characters are produced by non ASCII characters, eg those user defined, do not appear correctly. A pair of inverted commas is replaced by two single inverted commas. A more elaborate program would be required if a listing contains characters outside the normal ASCII range.

PROBLEMS

Some BBC machines do produce an unstable strobe line which does not maintain a high logic state and causes the PET to continue accepting the last character shown on the data bus. The solution, incorporated into later BBC models, is to solder a 2K2 resistor between pin 1 of the printer plug and pin 14 of IC 85 (A 5 volt source, ie high). This modification is best done on the underside of the printed circuit board.

The WAIT statement is line 6070 prevents breaking into the program by using the STOP key if this is the current line being executed. Since this is the condition on completion of printing, the solution to breaking into the BASIC program is to have it move from this line by printing from the BBC and interrupting during this subsequent print sequence.

Here is the transfer procedure:

M V Smith

1. Develop BBC program.
2. When hard copy is required connect PET and BBC using the cable in Fig. 1.
3. Load the PET transfer program and RUN. The program will hold on line 6070.
4. To enable the BBC printer output press Control and B. The command LIST will cause listing to both BBC monitor and PET printer.
5. To disenable the PET printer, type CONTROL and C.
6. To cause PRINT statements within the BBC program to print to the Commodore printer enable the printer prior to the statement with the command VDU 2 and disenable with VDU 3, eg

```
100  VDU 2
110  PRINT "FRED"
120  VDU 3
```

FURTHER APPLICATIONS

The transfer of programs from BBC to PET can be achieved by changing the device number in line 6030 from 4 to 3 and therefore cause listing to the screen of the PET. Care must be taken to ensure that the line numbers of that program to be transferred do not coincide with line numbers of the PET transfer program (it may be necessary to renumber the latter to avoid this).

Line **Description**

6020 Sets variables to speed up the BASIC.

6030 Opens file to the printer and sets the parallel user point to receive data.

6040 Causes a flag to be set on the negative transition of the input line CA1 (PIN B).

6050 Causes CB2 (PIN 19) to be an output line.

6070 Waits for the flag to be set by CA1 going low.

6080 Reads the data latched in on the transition of CA1 and in doing so resets the flag.

6090 Checks for lower case characters in the range 91-122.

6100 If a lower case letter is found then informs the printer (CHR (17)) and goes to line 6130 to print the ASCII for upper case.

6110 Looks for double inverted commas which causes inverted characters to be printed in strings. Overcomes the problem by going to line 6120.

6120 Prints two single inverted commas instead.

6130 Prints the character to the printer file.

6140 Defaults to upper case letters.

6150 Sets the acknowledge line low.

6160 Returns the acknowledge line to high.

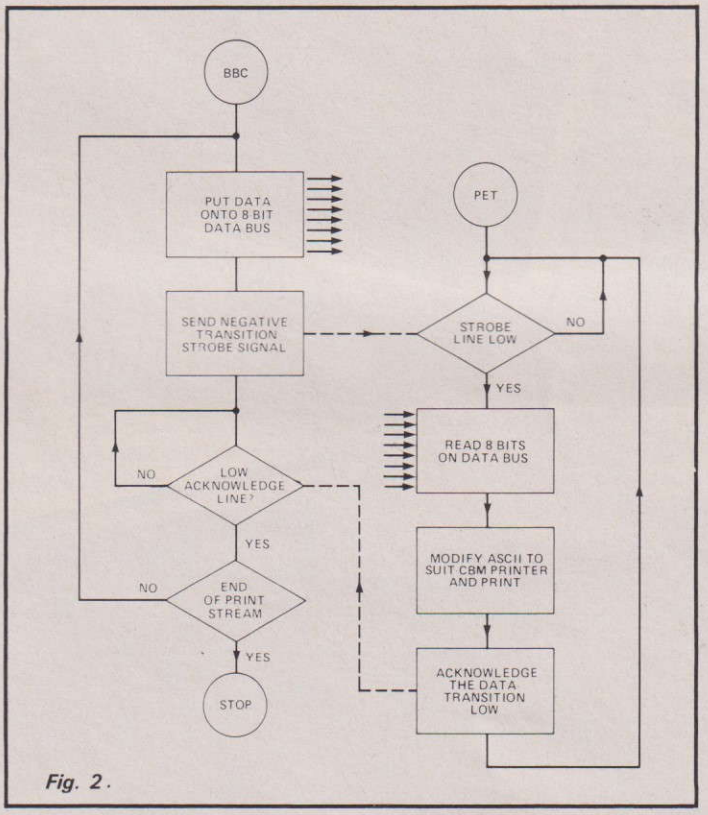


Fig. 2.

When three quarters of the screen has been listed then break into the PET program, HOME the cursor and insert with the Return key the new program lines. Then type 'RUN 6000' and continue to list another screen of program. When the complete BBC program has been transferred, it will need some alterations from the BBC to the PET BASIC together with deletion of lines 6000 forward.

CONCLUSION

Once again this project is only suggested as a temporary measure for the printer problem and the true solution must lie in purchasing a correct Centronics or Serial type of printer.

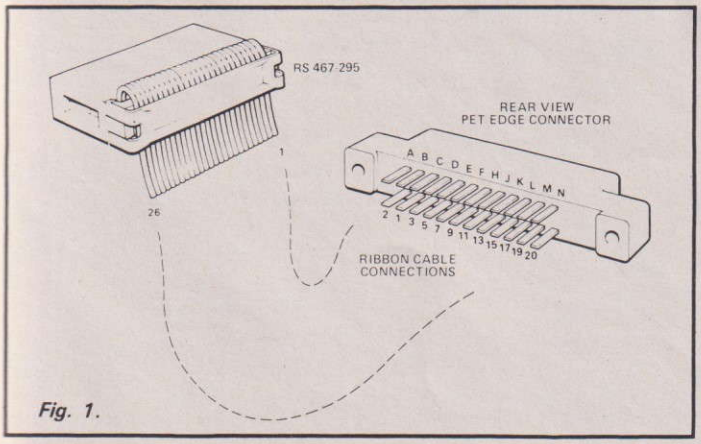
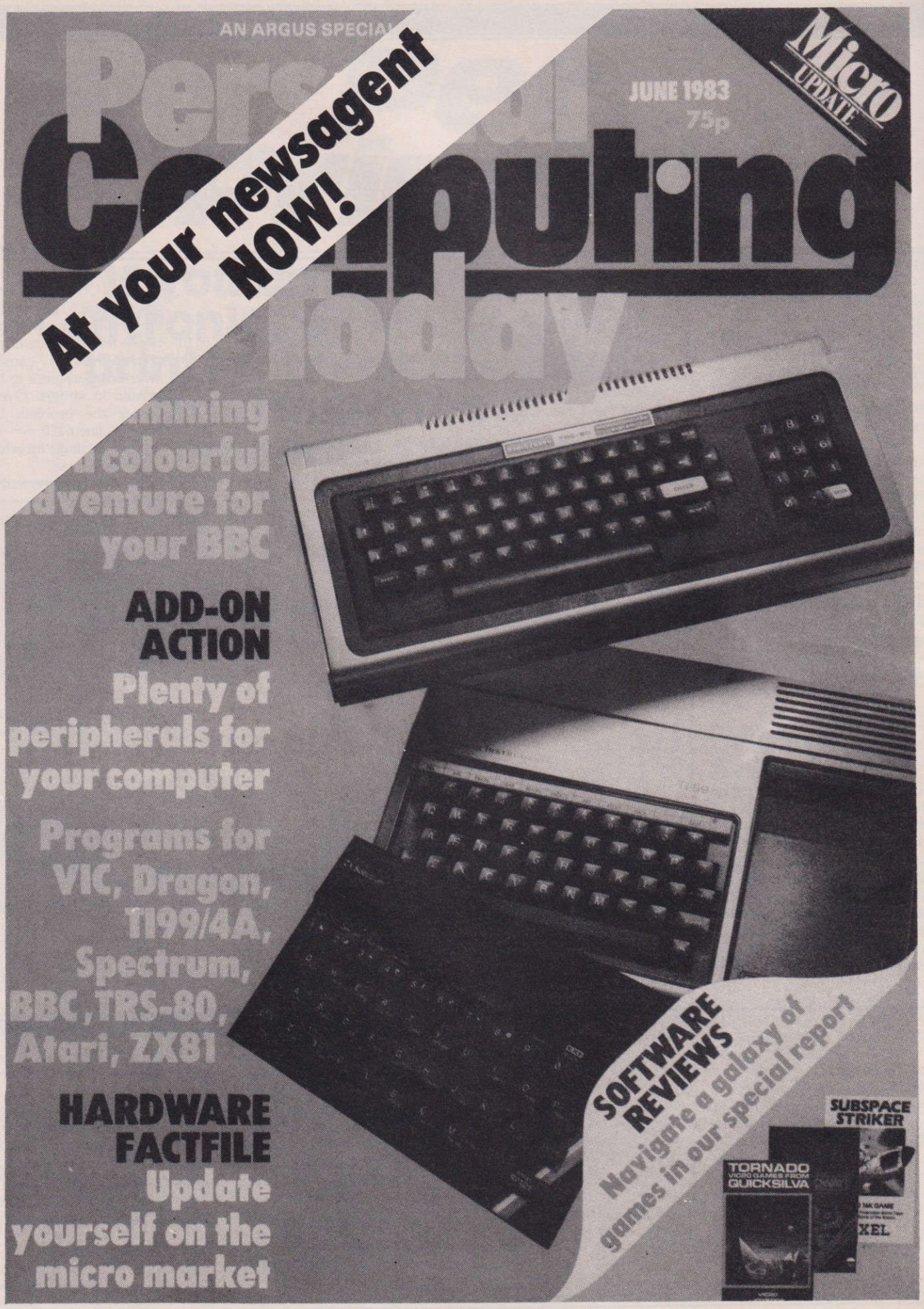


Fig. 1.

PROGRAM LISTING

```
6000 REM PETPRINTER-BBC
6010 REM.V.SMITH 12/11/82
6020 Z=34:X=91:Y=122:B=59468:C=32:D=59457:E=59469:S=39
6030 OPEN1,4:POKE59459,0
6040 POKEB,(PEEK(B)AND NOT1)
6050 POKEB,(PEEK(B)OR128OR64ORC)
6060 GOTO6080
6070 WAIT E,2
6080 A=PEEK(D)
6090 IFA<XORA>YTHEN6110
6100 A=A-C:PRINT#1,CHR$(17);:GOTO6130
6110 IF A<>ZTHEN 6130
6120 PRINT#1,CHR$(S);CHR$(S);:GOTO6150
6130 PRINT#1,CHR$(A);
6140 PRINT#1,CHR$(145);
6150 POKEB,(PEEK(B)AND NOTC)
6160 POKEB,(PEEK(B)ORC)
6170 GOTO6070
READY.
```


AN ARGUS SPECIAL
Personal Computing Today
JUNE 1983
75p
Micro UPDATE
At your newsagent NOW!
...mming a colourful ...venture for your BBC
ADD-ON ACTION
Plenty of peripherals for your computer
Programs for VIC, Dragon, TI99/4A, Spectrum, BBC, TRS-80, Atari, ZX81
HARDWARE FACTFILE
Update yourself on the micro market
SOFTWARE REVIEWS
Navigate a galaxy of games in our special report
SUBSPACE STRIKER
TORNADO
QUICKSILVA

Character Definer

K J O'Connell

Create your own graphics characters to bring a touch of imagination and originality to programs using this neat technique.

Defining characters can be a lengthy business, and if you are writing, say, a games program which uses a good many, you could be busy for hours using up a lot of paper and a lot of patience. However, this program takes out all the hard work, saving both time and tempers. It allows you to build up the character on an eight by eight grid. Then, simply by entering grid co-ordinates, it defines the character and prints it out on the screen as it is held in memory, while also giving the associated VDU statement. In addition, the program will also print composite characters.

To the veteran BBC Micro users who have had their machines for a fortnight or so, the subject of character definition probably holds few surprises. But for the new user, who still has bits of expanded polystyrene squeaking and snapping underfoot, perhaps a brief introduction may be necessary.

A LOAD OF CODE

Characters are addressed by number, as the ASCII code associated with that character, or as the hexadecimal value of that code.

The user-definable characters are, ostensibly, those between ASCII codes 224 and 255 (&E0 – &FFHex), 32 in all; in fact, the entire character set can be redefined, but space has to be set aside in memory first.

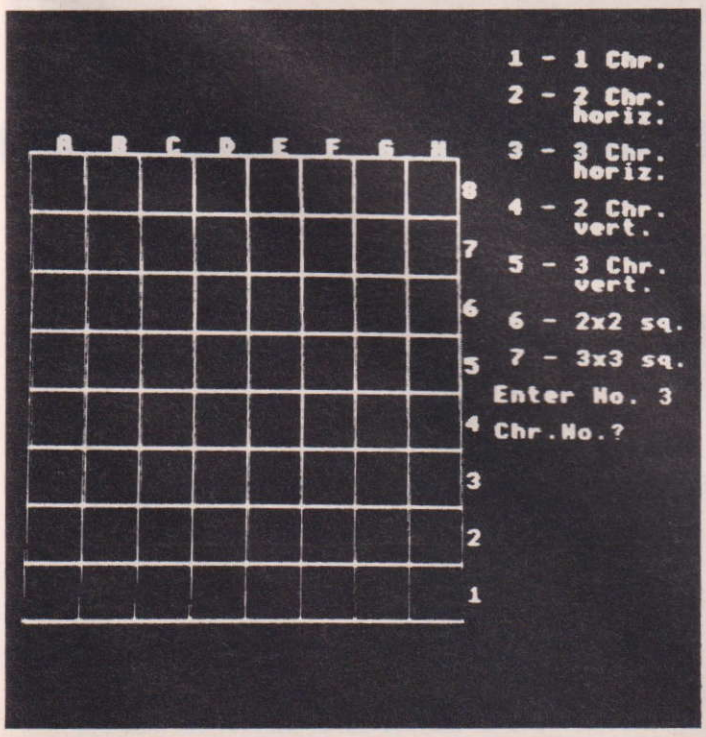
Characters are built up of an eight by eight matrix of dots and, therefore, to define a character you must plot it on an eight by eight grid. The vertical columns of the grid are numbered from right to left, beginning with 1, and doubling with each column thus: 1,2,4,8, etc, up to 128. These numbers denote the binary values of the squares. So, in any horizontal row, squares plotted have their values added together, the sum of which then represents the binary value of that row. Thus, if the first (upper) horizontal row has the second and sixth squares from the right plotted '--x---x-', the value would be 32+2=34; the 0s and 1s of the equivalent binary figure showing a direct relationship to the squares would be '00100010'.

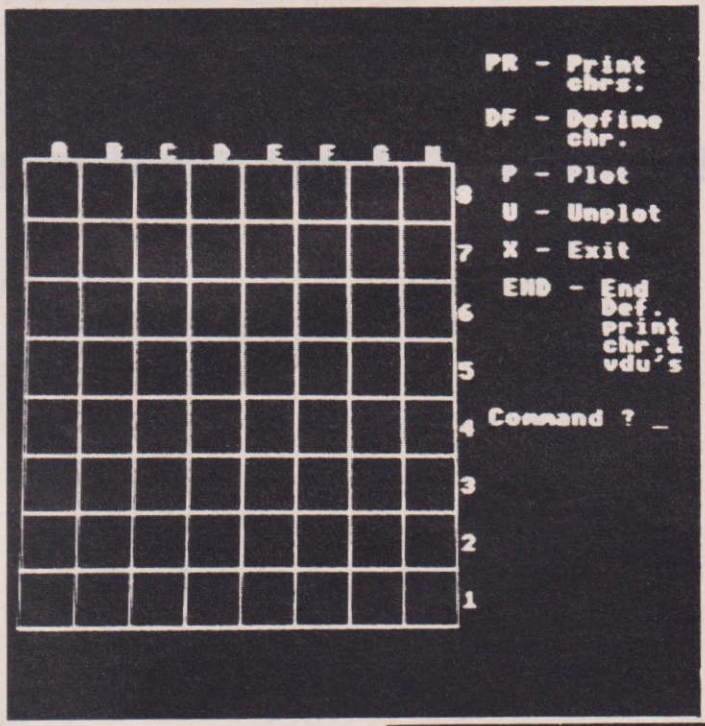
After plotting the character on the grid, it is entered using the character define command VDU 23, followed by the character number, ie 250, and the values of the eight horizontal rows. Thus, VDU 23,250,1,2,4,8,16,32,64,128 would define character 250 to be a diagonal line.

By plotting a number of characters adjoining each other, and using the characters 8,9,10, and 11 to reposition the cursor back, forward, down, and up, one character space respectively, larger, composite characters can be built up. For example:

```
PRINT CHR$(N);CHR$(N);
CHR$(8);CHR$(8);CHR$(10);
CHR$(N);CHR$(N)
```

would give a single character built up of four smaller ones arranged in a two by two square.





CONTINUED OVER

Then, if this were stored as a string, ie n$=Chr.(n)+Chr.(n)+Chr.(8). . . .etc, the complete character could be printed simply with "PRINT n$".

It would appear that by defining a number of composite characters, the available 32 could be used up very quickly. However, as the BBC Micro's screen is memory mapped, the same characters can be re-defined as many times as required within a program — once the character is printed on the screen, the screen memory holds it there and the original character can be re-defined.

SPOILT FOR CHOICE?

On RUNning the program, a large eight by eight grid, numbered like a chessboard, is displayed on the screen, together with a banner saying 'HL for Help', and the prompt 'Command?'. Entering HL followed by Return gives a menu, displayed in a narrow text column at the right-hand side of the screen, offering the following options:

P - Plot After entering this, the computer responds with the prompt 'Square?'; giving the co-ordinates of the square, ie B,4, would cause that square to be filled in on the grid. This is then followed by the prompt 'Command?'.

U - Unplot Similar to Plot except that the entered square is erased.

DF - Define This informs the computer that a character is to be defined — Plot commands by themselves are not held in memory, the grid at this stage simply a sketch pad. After entering DF, the computer gives the prompt 'Chr.No. ?', asking for the ASCII code of the character to be defined. After entering the character number, the associated character is flushed of all values and the prompt 'Command?' is given. The squares then being plotted employ the plot/unplot routines.

END - End Definition On entering END, the defined character is displayed in the text column, and the VDU statement is printed along the top if the screen.

X - Exit When entered in response to the 'Command?' prompt, allows exit from the program.

PR - Print Entering this command gives another menu, offering various formats for the printing of composite characters. This is then followed by the prompt 'Enter No.'. After entering the number for the print format desired, the prompt 'Chr.No.?' will appear. This will be repeated until all the characters for a given format have been entered, after which the composite character will be displayed in the text column. The entire character set is accessed by PR in addition to the user-defined characters. The menu offered by PR is shown below.

1 — Print a single character.
2 — Print two characters in a row.
3 — Print three characters in a row.
4 — Print two characters in a vertical column.
5 — Print three characters in a vertical column.
6 — Print four characters in a two by two square.
7 — Print nine characters in a three by three square.

IN BRIEF

The grid is in effect bit-mapped into the integer variables, A% to H%, between lines 2030 and 2100 in PROCdefine, the value of Y% determining the row into which the values (representing the plotted squares for that row) carried by J% are to be fed. Q% is normally at 0; it only takes on the value of J% during the unplotting routine when it is used to subtract the value of the erased square.

The value of a plotted square is related to its distance from the graphics origin, ie from the bottom left-hand corner, whereas the values of a defined character need to be read from right to left and top to bottom. The expression at line 1990, therefore, prevents the production of a mirror image by reversing the order of the value of X%.

PROCchr is concerned with print format; characters inputted are entered into strings to be printed out as composites after first going through PROCcheck to ensure that control codes, ie , 32, are not being accessed.

PROGRAM LISTING

```
  10 MODE 4
  20 VDU 28,27,31,39,0
  30 DIM A(9)
  40 A%=0:B%=0:C%=0:D%=0:E%=0:F%=0:G%=0:H%=0:J%=0:Q%=0:U%=0:V%=0:W%=0:Z
%=0
  50 I%=32:K%=32:L%=32:N%=32:O%=32:P%=32:R%=32:S%=32:T%=32
  60 CLG
  70 PROChead
  80 PROCscreen
  90 PROCcommand
 100 IFW%=0 OR C$="X" THEN 210
 110 VDU 23,W%,H%,G%,F%,E%,D%,C%,B%,A%
 120 CLS
 130 VDU 5
 140 MOVE 10,950
 150 PRINT"VDU 23,";W%,";H%",";G%",";F%",";E%",";D%",";C%",";B%",";A%
 160 VDU 4
 170 PRINT TAB(0,6);"chr.";W% now -"
 180 PRINT' SPC(6);CHR$W%
 190 PRINT' ; INPUT" finished Y/N",R$
 200 IF R$="N" THEN 40
 210 MODE 4
 220 END
 230
 240
 250 DEF PROCcommand
 260 PRINT':INPUT "Command ? "C$
 270 IF C$="PR" OR C$="DF" OR C$="P" OR C$="U" OR C$="HL" OR C$="X"OR C
$="END" THEN 290
 280 PRINT' "unrecognized command": GOTO 260
 290 IF C$="X" OR C$="END" THEN 480
 300 IF C$="PR" THEN PROCformat
 310 IF Z%>0 THEN PROCchr
 320 IF C$="HL" THEN PROChelp
 330 IF C$<>"DF" THEN 440
 340 CLG
 350 PROChead
 360 PROCscreen
 370 PRINT:INPUT "Chr. No.",W%
 380 IF W%<224 OR W%>255 THEN 390 ELSE 430
 390 PRINT' " Definable"
 400 PRINT" Chrs. are"
 410 PRINT" 224 - 255"
 420 GOTO 370
 430 VDU 23,W%,0,0,0,0,0,0,0,0
 440 IF C$="P" THEN P%=85
 450 IF C$="U" THEN P%=87
 460 IF C$="P" OR C$="U" PROCplot
 470 GOTO 260
 480 ENDPROC
 490
 500
 510 DEF PROCplot
 520 PRINT: INPUT "square ? "X$,Y
 530 IF LEN(X$)<>1 THEN 590
 540 IF ASC(X$)<65 OR ASC(X$)>72 THEN 590
```

```
550 IF Y<1 OR Y>8 THEN 590
560 Y%=(Y-1)*100
570 X%=(ASC(X$)-65)*100
580 PROCfill : GOTO 650
590 PRINT' "co-ords must be entered asa number and a letter"
600 PRINT "separated by a comma."
610 PRINT' "eg. A,4 <RET>"
620 PRINT "or as number and letter"
630 PRINT "separated by Return :-"
640 PRINT' "A<RET> 4<RET>"
650 ENDPROC
660
670
680 DEF PROChead
690 VDU 5
700 MOVE 20,1020
710 PRINT"HL for Help"
720 VDU 4
730 ENDPROC
740
750
760 DEF PROCcheck
770 IF A(LOOP)>31 THEN 810
780 PRINT' Chrs<32 are"
790 PRINT" control"
800 PRINT" codes.":GOTO 830
810 IF A(LOOP)<256 THEN 850
820 PRINT' " 255 is max."
830 PRINT'" Restart"
840 PRINT" format ";V% : U%=1
850 ENDPROC
860
870
880 DEF PROChelp
890 CLS
900 PRINT' "PR - Print"
910 PRINT TAB(5);"chrs."
920 PRINT' "DF - Define"
930 PRINT TAB(5);"chr."
940 PRINT' " P - Plot"
950 PRINT' " U - Unplot"
960 PRINT' " X - Exit"
970 PRINT' " END - End"
980 PRINT TAB(7);"Def."
990 PRINT TAB(7);"print"
1000 PRINT TAB(7);"chr.&"
1010 PRINT TAB(7);"vdu's"
1020 ENDPROC
1030
1040
1050 DEF PROCformat
1060 CLS
1070 PRINT' " 1 - 1 Chr."
1080 PRINT' " 2 - 2 Chr."
1090 PRINT TAB(5);"horiz."
1100 PRINT' " 3 - 3 Chr."
1110 PRINT TAB(5);"horiz."
1120 PRINT' ' 4 - 2 Chr."
1130 PRINT TAB(5);"vert."
1140 PRINT' " 5 - 3 Chr."
1150 PRINT TAB(5);"vert."
1160 PRINT' " 6 - 2x2 sq."
1170 PRINT' " 7 - 3x3 sq."
1180 PRINT : INPUT "Enter No. "Z% : V%=Z%
1190 IF Z%<1 OR Z%>7 THEN 1180
1200 IF Z%=7 THEN Z%=9 : GOTO 1220
1210 IF Z%>3 THEN Z%=Z%-2
1220 ENDPROC
1230
1240
1250 DEF PROCchr
1260 U%=0
1270 LOOP=1
1280 REPEAT
1290 PRINT : INPUT "Chr.No.? "A(LOOP) : PROCcheck
1300 IF U%=1 THEN LOOP=1 : GOTO 1260
1310 LOOP=LOOP+1
1320 UNTIL LOOP=Z%+1
1330 Z%=0
1340 AA$=CHR$A(1)+CHR$A(2)
1350 AB$=AA$+CHR$A(3)
1360 BB$=CHR$A(1)+CHR$8+CHR$10+CHR$A(2)
1370 BC$=BB$+CHR$8+CHR$10+CHR$A(3)
1380 CC$=CHR$A(3)+CHR$A(4)
1390 CD$=CHR$8+CHR$8+CHR$10
1400 DD$=CHR$A(4)+CHR$A(5)+CHR$A(6)+CHR$8+CD$
1410 DE$=CHR$A(7)+CHR$A(8)+CHR$A(9)
1420 PRINT
1430 IF V%=1 THEN PRINT' TAB(4);CHR$A(1)
1440 IF V%=2 THEN PRINT' TAB(4);AA$
1450 IF V%=3 THEN PRINT' TAB(4);AB$
1460 IF V%=4 THEN PRINT' TAB(4);BB$
1470 IF V%=5 THEN PRINT' TAB(4);BC$
1480 IF V%=6 THEN PRINT' TAB(4);AA$;CD$;CC$
1490 IF V%=7 THEN PRINT' TAB(4);AB$;CHR$8;CD$;DD$;DE$
1500 ENDPROC
1510
1520
1530 DEF PROCscreen
1540 X=0
1550 FOR Y=0 TO 800 STEP 100
1560 MOVE X,Y
1570 DRAW X+800,Y
1580 NEXT
1590 FOR Y=0 TO 800 STEP 100
1600 MOVE Y,X
1610 DRAW Y,X+800
1620 NEXT
1630 VDU 5
1640 B=830:C=0
1650 FOR A=50 TO 750 STEP 100
1660 MOVE A,B
1670 PRINT CHR$(&41+C)
1680 C=C+1
1690 NEXT
1700 C=0
1710 FOR A=60 TO 760 STEP 100
1720 MOVE B-20,A
1730 PRINT CHR$(&31+C)
1740 C=C+1
1750 NEXT
1760 VDU 4
1770 ENDPROC
1780
1790
1800 DEF PROCfill
1810 MOVE X%,Y%
1820 IF P%=85 AND POINT(X%+10,Y%+10)=1 THEN 1920
1830 DRAW X%+100,Y%
1840 DRAW X%+100,Y%+100
1850 PLOT P%,X%,Y%
1860 MOVE X%,Y%
1870 DRAW X%,Y%+100
1880 DRAW X%+100,Y%+100
1890 PLOT P%,X%,Y%
1900 IF P%=85 THEN 1910 ELSE PROCscreen
1910 PROCdefine : GOTO 1940
1920 PRINT' " allready"
1930 PRINT" plotted"
1940 ENDPROC
1950
1960
1970 DEF PROCdefine
1980 X%=X%/100 : Y%=(Y%/100)+1
1990 X%=ABS(X%-7)
2000 J%=2^X%
2010 IF P%=85 THEN 2030
2020 Q%=J% : J%=J%-(2^X%)
2030 IF Y%=1 THEN A%=(A%+J%)-Q%
2040 IF Y%=2 THEN B%=(B%+J%)-Q%
2050 IF Y%=3 THEN C%=(C%+J%)-Q%
2060 IF Y%=4 THEN D%=(D%+J%)-Q%
2070 IF Y%=5 THEN E%=(E%+J%)-Q%
2080 IF Y%=6 THEN F%=(F%+J%)-Q%
2090 IF Y%=7 THEN G%=(G%+J%)-Q%
2100 IF Y%=8 THEN H%=(H%+J%)-Q%
2110 Q%=0
2120 ENDPROC
```

News Ne

THE TOOLS OF THE TRADE

Following the success of 'r q FORTH' for the BBC Micro, Level 9 Computing have released a new software package called FORTH TOOLKIT. Adding over 200 new functions to their previous release, FORTH TOOLKIT comes on cassette, complete with a 60 page manual and summary card, for £10 inclusive.

The new toolkit provides a full 6502 assembler, which you can use for machine code within your FORTH programs; turtle graphics, which provide simple-to-use colour graphics routines; location, comprising four utilities to help find where words are used; decompiler utilities, comprising five versatile routines for examining compiled FORTH programs; double numbers, providing 25 double-number operations in addition to those of 'r q FORTH' itself; cassette file handling, providing easy use of data files on cassette; printer and joystick handling; and bit pattern manipulation, comprising routines to shift and rotate bits within a number.

The package is supplied in the form of several separate modules so you only need to load the parts of the toolkit that you want to use at any particular time.

For further details of the package, FORTH TOOLKIT, contact Level 9 Computing, 229 Hughenden Road, High Wycombe, Bucks HP13 5PG or 'phone 0494 26871.

TODAY THE UNIVERSE, TOMORROW . . .

Acorn Computers are taking part in Project Universe, a £3 million three year research study to assess the future and practical problems of linking computers over long distances at high speed.

The BBC Micro, here linked to the Acorn Dual Disk Drive which has just come down in price by more than £100.

Acorn's subsidiary, Orbis Computers, have developed a special interface, which allows the BBC Micro to be connected in the Cambridge Ring local area network via a node as part of this project. In the Cambridge Ring, packets of digital information circulate around the ring at a data transmission rate of IM per second. Orbis have developed their system to allow the BBC Micro to be used as an interactive graphics terminal and it is currently being used by scientists at University College London and Loughborough University.

Mr Peter O'Keefe, the General Manager of Orbis, states "This now demonstrates that the BBC Micro is capable of being connected to three different levels of network, namely the high speed Cambridge Ring, Acorn's own Econet Lan, primarily used in educational computing, and over the telephone line via an RS232 link".

A SLIPPED DISC?

When was the last time you heard of something coming down in price? Well, Acorn have just announed that the cost of their 800K discs are being reduced by more than £100.

Currently priced at £918.85, the Acorn Dual Disk Drive unit will be featured in the company's new catalogue at only £803.85. Their reason for the price reduction is because they are selling so many that they are able to cut their costs. However, there are not any more price cuts expected no matter how many people take advantage of this offer, so this may be a good time to invest in a disc system.

At present, most owners of the Acorn Dual Disk Drive unit are in schools or in small business.

GET GRAPHICAL

Nobody, but nobody, could accuse the BBC Micro of not being able to produce superb graphics, but it is a sad fact that many of you find the various Move, Plot and Draw options just a little confusing. So, what can you do to make things easier?

One of the options available is to invest in a graphics package which provides the would-be artist with a host of pre-defined functions and features at his or her fingertips. One of the latest of these packages comes from Salamander Software who

ws News

recently acquired the rights to market it in the UK. Originally written by EDG Engineering International, a firm of consultants and design engineers, the package has seen use in the commercial environment before going public.

Running on the Model B system, the cassette based program requires no additional facilities as it stores its completed drawings on tape and uses the keyboard to control the movement of the on-screen cursor. Drawings are created in Modes 0,1 and 2 and the user has access to pre-defined shapes such as circles and boxes which may be solid, shaded or filled. Future expansion of the system inlcudes plans to offer a disc based version with joystick or paddle input and graphics tablets as the input devices with hard copy printout of the designs also being available.

The current package is selling for £24.95 inclusive and is available from Salamander Software at 17 Norfolk Road, Brighton, Sussex BN1 3AA or directly from your local dealer.

ACORN SOLDER ON

If you have any problem with the memory chips inside your BBC Micro, you'll find you've got a major problem on your hands.

When Acorn were putting the machine together, they soldered the memory chips permanently into place on the main board. This leads to an obvious difficulty should you get a problem chip—you won't be able to deal with this snag yourself, unless, of course, you've had a lot of experience with this type of problem.

The official word from Acorn is that the chips were originally soldered on to make the BBC Micro's operation more reliable, and also to allow the machine to be easily transported without the chips coming loose. It was added that RAM chips don't fail anyway, but this is hardly reassuring to anyone with experience of computers!

If you think you are having a problem which could be related to a faulty chip, return the machine to your nearest Acorn service centre and let them deal with it.

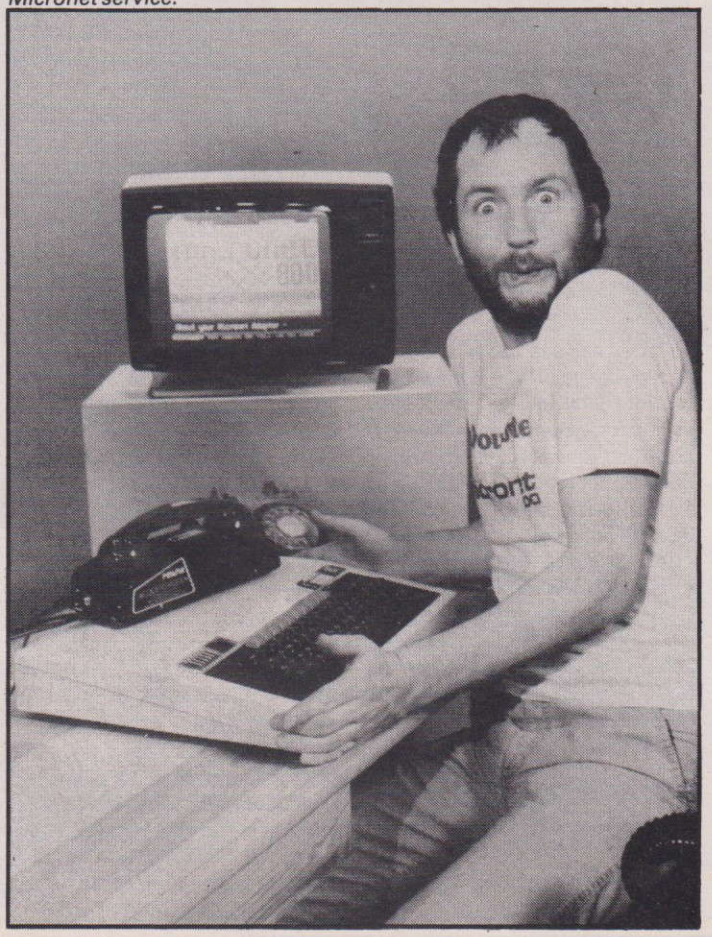
SOFTWARE SPECIAL

When Micronet organiser, Telemap, decided to find the minimum 100 programs for their subscribers, they had little difficulty filling the spec for the Commodore machines, Apple II, the Tandy models I and III, and RML380Z. But, when they turned their attention to the BBC Micro, their troubles began—there just wasn't enough copyright-free software material for their needs.

So, not wishing to disappoint their subscribers, Telemap have commissioned a software house to convert 25 programs from the public domain.

However, as most of the programs newly converted for the BBC Micro originally ran on the Commodore PET and VIC-20, there have been some cross words from the Commodore Users Group, IPUG. They feel that the BBC Micro should have been seen to be unsupported with software.

Caught in the act – Cuddly Ken gets some hands-on experience of the Micronet service.



BECOME A BARD

Fancy having a piece of software tailored to fit your requirements? Well, if you are into writing or the production of reports, letters and other lengthy documents you'll hardly need to tell me that a word processing package is a must. So, what's new about this one then?

Wordsworth (that's the name of the package) has been produced for the BBC Micro with the direct intention of allowing it to be adaptable to suit any of the various Operating Systems or, indeed, any of the currently popular printers. And, at only £20 all inclusive, that has just got to be a bargain. The basic package is configured for the Epson MX-80 Type III printer but the author is more than willing to adapt it to suit your particular peripheral.

Features implemented include Save, Load and Merge of documents from tape, the ability to copy, move or delete sections (you can even recover something that you just deleted by mistake) and a host of other useful facilities. Among the more unusual offerings are a word count, essential for many users but seldom supplied, and a lock function which allows others to read but not alter text stored in the machine.

If you fancy spreading the word all over the page contact Ian Copestake at 23 Connaught Crescent, Brookwood, Woking, Surrey GU24 0AN.

BI-LINGUAL BBC

Adding a second language to your BBC Micro could hardly be easier these days, especially with the independent software houses entering the market in a big way.

Latest on the scene is HCCS

CONTINUED OVER

News Ne

Associates who will supply you with FORTH in a ROM for just £34.72 (plus VAT). Designed to work with all current versions of operating system and to be fully compatible with the graphics and expansion facialities, something that many cassette based FORTHs are not, the language comes complete with all necessary documentation. For the beginner in FORTH, however, an extra book is available entitled **Welcome FORTH** which costs an extra £6.75.

Orders can be supplied directly from HCCS at 533 Durham Road, Low Fell, Gateshead, Tyne and Wear NE9 5EY or you can telephone 0632-821924 for further information.

DEAR AUNTIE . . .

So many people have written to Acorn regarding the BBC Micro that they have decided to set up a special service to cope with the correspondence.

In much the same way as the BBC service, there is no telephone number—but as I'm sure you all know it is almost impossible to talk to anyone on the 'phone about the BBC Micro!

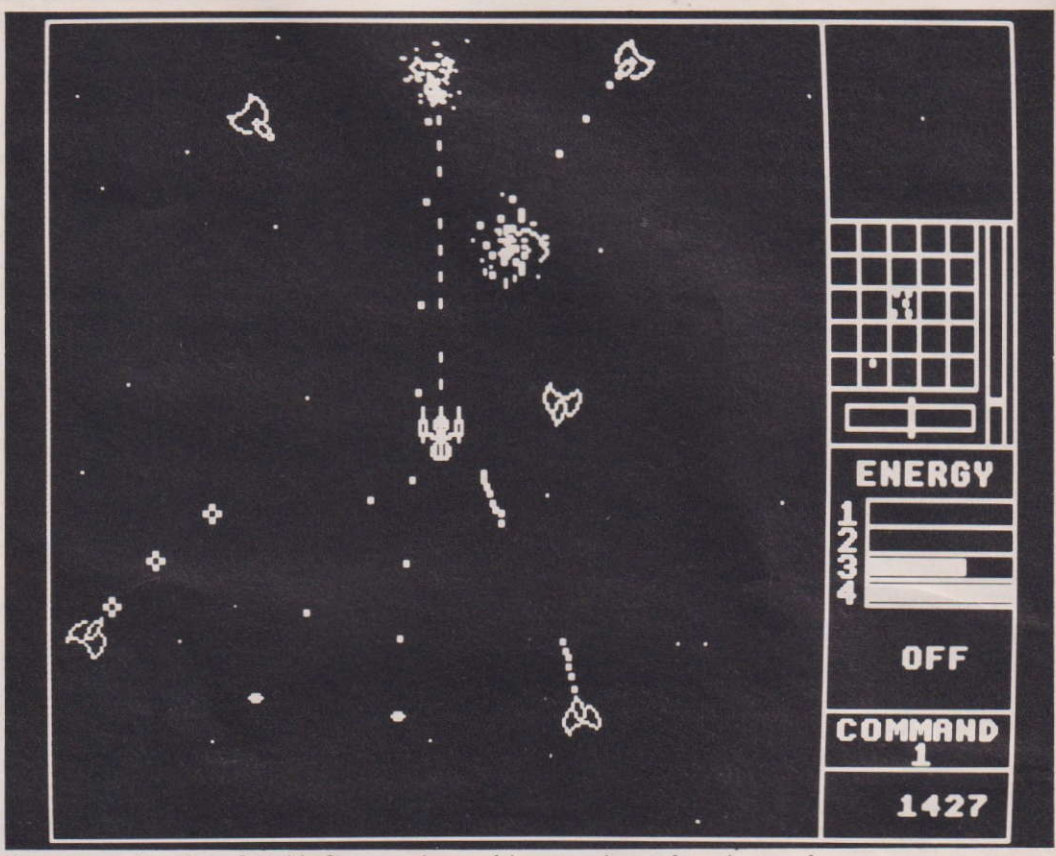
Written queries will be dealt with as soon as possible by Mike Bicknall of the Customer Service Centre, Acorn Computers, Fulborne Road, Cherry Hilton, Cambridge CB1 4JN. Don't be too impatient though—they have a very busy job.

TEN THOUSAND TIMES!

If you were ever worried that your BBC Micro wouldn't be big enough to store all the information you need, how does 340M of Winchester storage sound? Using a new interface system produced by Andover based Geophysical Systems you can now connect up to four 85M Winchester discs to your Model B along with the option of mass tape backup as well.

GSL will sell the whole thing as a package or you can just buy the interface and as many discs as you feel you need. Software to control this massive amount of data is currently available as a simple utility program, but this will be extended and translated into machine code for eventual delivery in ROM format.

For more information on how to make you BBC Micro's memory 10,000 times bigger contact GSL at 2 North Way, Andover, Hampshire SP10 5AZ or ring on 0264-58744.



On-screen action from Starship Command, one of the new releases from Acronsoft.

COMMAND CONTROL

Coming soon from Software supplier Acornsoft are three exciting new games which push the BBC Micro to its limits.

The first, Starship Command, is a variation of the tried and trusted 'blast the aliens' game. In it you control a powerful space fighter (which looks surprisingly like the Liberator from the popular space fiction TV programme, Blake's Seven) trying to escape from an alien infested sector of space. The whole game is run in real-time with some pretty impressive graphics and requires a lot more than a quick 'trigger-finger' to escape alive.

The second tape is entitled Missile Command. As you might have guessed, this is a computer version of the much-loved arcade game of the same name. The programmers at Acornsoft have worked hard at this one and have come up with a faithful reproduction which does full justice to its original arcade counterpart.

If you have tried killing and maiming aliens, you might like to check out the third Acornsoft tape. Called Snooker, it brings alive the skill and dexterity of the real game in the comfort of your own home.

The realism of the game is enhanced with features such as top and back spin, thus increasing the control you have over your shots. The graphics and presentation of the game are up to the usual high standards of the other Acornsoft tapes, even to the point of displaying the scores on a proper scoring rack.

Starship Command, Missile Command and Snooker are all priced at £9.95. For further

News News

information contact Acornsoft, 4a Market Hill, Cambridge or telephone 0223 316039.

BASIC II

Despite rumours to the contrary it seems there is a second version of BASIC being supplied with some of the new BBC computers. BASIC II, as it is referred to, is fitted to machines with a 1.2 ROM operating system.

There is an easy way to find out whether or not your machine has this BASIC fitted. On power-up, or after pressing the Break key, type REPORT and press Return. The machine will then reply with a copyright notice. If the notice shows '1981' then the machine has the original BASIC fitted; if, however, it shows '1982' then it has BASIC II.

Before you rush to your dealer and demand the new BASIC, it is worth noting that there is no great difference between the two. BASIC II features a few extra commands for the assembler and some software to help with passing command lines to the operating system. This new BASIC will only be of use to the dedicated programmer. So, for the vast majority of BBC BASIC users there is no need to upgrade your machine.

NETWORKING TAKES OFF

The BBC Micro's Econet networking system has been recommended for use in 200 Information Technology Centres across the country. These centres, which are funded by the Department of Industry and the Manpower Services Commission, have been set up to train school leavers in computing skills.

This announcement coincides with the release of Acorn's February sales figures which show slightly under a quarter of all BBC Micros produced each month are fitted with an Econet interface. The bulk of these machines have been supplied to the education market, the rest finding their way into large establishments such as British Telecom.

The Econet system allows BBC Micros to be linked together, thus enabling systems such as electronic mail and data collection to be set up using a stand alone micro. Perhaps the biggest asset is for schools because the Econet system will allow many computers to share peripherals such as discs and printers. It also enables teachers to 'eavesdrop' on their pupil's progress. I'm glad they didn't have that in my day!

The Econet interface costs about £50.00 per micro. Further information about this networking system can be obtained from Peter O'Keefe, General Manager, Networking Division, Acorn Computers Limited, Fulbourn Road, Cherry Hinton, Cambridge CB1 4JN. Telephone enquires can be made on 0223 312449.

BBC Disc Director

Make more of the BBC's disc drive system when amassing data files. We show you how to overcome some existing inadequacies.

The BBC Microcomputer's disc filing system is simple but highly effective in most applications. There are several situations, however, where the current set of DFS commands are inadequate.

Specifically, many disc orientated programs will require the user to enter a filename at some point. A 'non user friendly' program simply prints 'Enter filename': and awaits your entry. The problem with this format is that if the user has forgotten what files are on the disc, he or she will be totally flummoxed at this point, and will have to exit the program in order to take a catalogue of the disc. This could mean the user loses any results generated by the program.

The solution, of course, is to present the user with a disc catalogue before any filenames are requested.

A BETTER SELECTION

Acorn's DFS has a command, *CAT, which does just that, but it still is not suitable in a number of instances. A typical disc catalogue is shown in Fig. 1. As you can see, the computer is not at all selective in the files it lists—if the program '+.CSTRCT' were requesting a filename, '+.CSTRCT' itself would appear as a candidate!

A second problem is the mass of information presented. When you are simply trying to remember filenames you do not need the top two lines of information, since they do not affect the filename used.

Ideally, a selective catalogue is required. This is not easy to get. A command called *INFO is provided which can list the names of a group of files, but it also presents masses of other information as well, so is not really useful in this situation,

```
>*CAT
BBC Revealed (33)
Drive 0          Option 3 (EXEC)
Directory :0.$   Library :0.$

     !BOOT     L       CAT
     FRAME1    L       FRAME2    L
     FRAME3    L       FRAME4    L
     FRAME5    L       OLDMAZE   L
     SCRNDMP   L

   +.3D-PLOT   L     +.3D-SIM    L
   +.3D-TRIG   L     +.6502DIS   L
   +.CIRCLES   L     +.CSTRCT    L
   +.GCALC     L     +.KEYLIST   L
   +.MOIRE     L     +.REVERSE   L
   +.SALVAGE   L     +.TABLES    L
>
>
```

Fig. 1 : A typical disc catalogue.

A second problem is that the standard catalogue always appears in the format illustrated—if you do a disc catalogue in Mode 0 (where there are 80 characters to a line), the filenames are still formatted to 40 columns.

The final answer is provided in Listing 1. Called as a function, it will fill a string array with all the filenames on a disc. The string array can then be processed however you like—for example, you could suppress the printing of all files in a certain directory or those which are locked.

The format of each element of the array is shown :

1	Directory
2	A full stop (this could be suppressed).
3 to 9	Filename (padded to the right with spaces).
10	Space.
11	'L' if the file is locked, space otherwise.

```
>RUN
$.CAT
$.!BOOT    L
+.REVERSE  L
+.MOIRE    L
+.GCALC    L
+.CSTRCT   L
+.3D-TRIG  L
+.3D-PLOT  L
+.TABLES   L
+.SALVAGE  L
+.KEYLIST  L
+.CIRCLES  L
+.6502DIS  L
+.3D-SIM   L
$.SCRNDMP  L
$.OLDMAZE  L
$.FRAME1   L
$.FRAME2   L
$.FRAME3   L
$.FRAME4   L
$.FRAME5   L
```

Fig. 2.

An unprocessed example of the function's output appears in Fig. 3.

BEST OF THREE

Before the function can be used, three arrays must be dimensioned. In the example, this is done in line 50. BUFFER% is a byte array of length 256 bytes (ie from 0 to 255). PARAM% is another byte array, this time of length 100. (See page 236 of the new **User Guide** for information on byte arrays). This length for PARAM% is far too generous—if your memory is tight, cut it to

J Ruston

about 20. CAT$ is a normal string array.

The value of the function is the number of files on the disc. This is shown in lines 60 to 100 of the sample program. The parameters of the function are:

1 The drive to be interrogated.
2 The variable BUFFER%
3 The variable PARAM%

The internal operation of the function is quite complex, and so need not be examined by the casual user. It operates by loading the first two sectors of the disc into memory and extracting the catalogue information from them. To understand how it achieves this, the disc format used by the BBC Micro must be known.

It is laid out towards the end of the **Disc System User Guide** but the salient points are shown in Figs. 3 and 4.

SECTOR 00

Bytes	Contents
&00–&07	First 8 characters of disc title.
&08–&0E	First filename
&0F	Directory of first filename.
&10–&1E	Second filename.
&1F	Directory of second filename.

etc, for up to 31 files

Fig. 3.

The BBC disc drives make a compact mass storage system for the BBC Micro.

CONTINUED OVER

SECTOR 01

Bytes	Contents
&00-&03	Rest of disc title.
&04	Update number.
&05	The number of catalogue entries multiplied by eight.

Fig. 4.

OFFSET

0	Drive number.
1-4	The destination address of the sectors.
5	3.
6	&53.
7	0 or 1, depending which sector is to be read.
8	&21.

Fig. 5.

PROGRAM LISTING

```
 50 DIM BUFFER% 255,PARAM% 100,CAT$(32)
 60 L%=FNcat(0,BUFFER%,PARAM%)
 70 IF L%=0 THEN PRINT "No files":END
 80 FOR FILE%=1 TO L%
 90 PRINT CAT$(FILE%)
100 NEXT FILE%
110 END
120
130 DEF FNcat(DRIVE%,BUFFER%,X%)
140 LOCAL Y%,L%,FILE%,T%,A$,DIR%
150 Y%=X% DIV 256
160 A%=&7F
170 ?X%=DRIVE%
180 X%!1=BUFFER%
190 X%?5=3
200 X%?6=&53
210 X%?7=0
220 X%?8=1
230 X%?9=&21
240 X%?10=0
250 CALL &FFF1
260 IF X%?10 THEN PRINT "Can't read catalogue (1)":END
270 L%=(BUFFER%?5) DIV 8
280 IF L%=0 THEN=0
290 X%?8=0
300 X%?10=0
310 CALL &FFF1
320 IF X%?10 THEN PRINT "Can't read catalogue (0)":END
330 FOR FILE%=1 TO L%
340 DIR%=?(BUFFER%+FILE%*8+7)
350 A$=CHR$(DIR% MOD 128)+"."
360 FOR T%=0 TO 6
370 A$=A$+CHR$(?(BUFFER%+FILE%*8+T%))
380 NEXT T%
390 IF (DIR% AND 128) THEN A$=A$+" L" ELSE A$=A$+"  "
400 CAT$(FILE%)=A$
410 NEXT FILE%
420=L%
```

The rest of the sector 01 need not concern us here.

Thus, the first thing the function does is to examine the fifth byte of sector 01, to establish how many files exist on the disc. Unfortunately, this process is unnaturally complex.

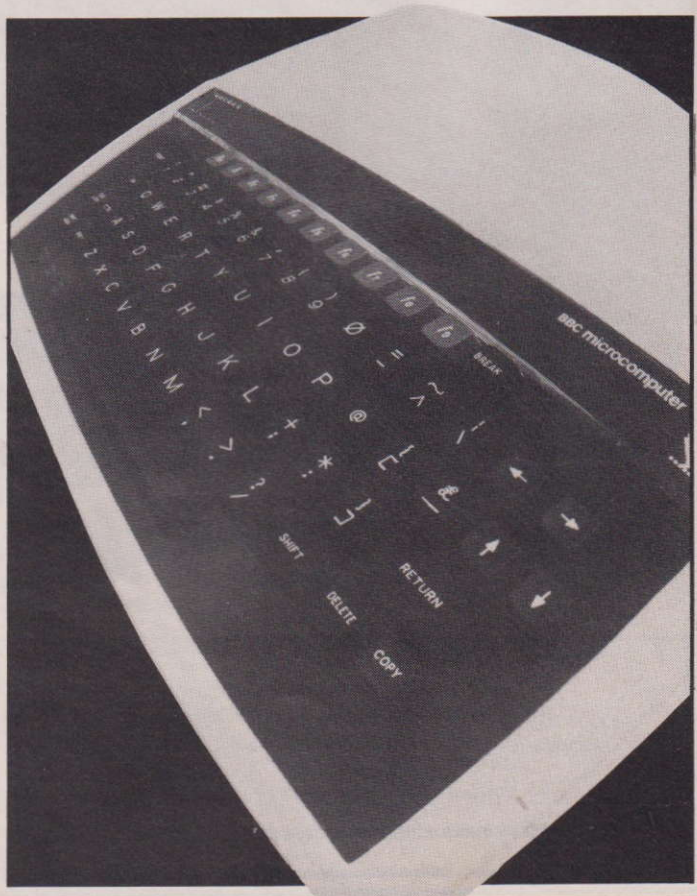
READ ALL ABOUT IT

To read a particular sector of a disc, a call to location &FFF1 Hex must be made, with the accumulator containing &7F Hex. In addition, the 6502's X (lsb) and Y (msb) registers must point to a free area of memory (actually PARAM%) for the parameters required for by the sector read routine. All this can easily be accomplished from BASIC.

The area of memory pointed to by X and Y contains the following information (and this is simplified . . .) shown in Fig. 5. (To see the above information in more detail, refer to the **Disc System User Guide**.)

In addition, if (XY)+10 does not contain zero after the sector has been read, an error has occured. This is signalled in lines 260 and 320.

The **Disc System User Guide** does not explain how locked (protected) files are indicated in the catalogue. In fact, all locked files have the top bit of the directory byte set to one. This is taken into account in lines 350 and 390.



Simple Pattern Animation

Disney did it with Mickey Mouse you can do it with our animation program.

Using the colour graphics command of the BBC Micro, it is possible to specify the colour of a line or point to be drawn with a number from 0 to 15. Unlike other colour computers, however, it is also possible to make each of these numbers correspond to any actual colour, or even all to the same colour. Thus, by drawing a pattern in 16 different colours and specifying all of these colours to be black, and then sequentially specifying each colour to come into view and the previous one to disappear, it is possible to create the illusion of movement. (For a further example of this technique, see the 3-D Animation feature in the December issue of **Computing Today**.) The advantage of this technique over others is that it is extremely rapid, and can affect as large an area of the screen as you please without any loss of speed.

The first example draws a cross on the screen in white and then rotates it at very high speed. If you think that's good, try the second, rather more spectacular, example.

No attempt has been made to explain the operation of the second example as it is rather more involved than the first, However, the principle is the same; a point to note is that in the second example, it is in lines 200 to 220 that the rotation is performed.

So put your talent to good use and key it all in.

PROGRAM LISTING 1

```
100  MODE 2
101  REM ** SET 16-COLOUR MODE
110  FOR I=0 TO 15
120  VDU 19,I,0,0;0
130  NEXT I
131  REM ** SET ALL COLOURS TO BLACK
140  Radius=500:J=1
141  REM ** J IS A COUNTER FOR PLOTTING
     COLOUR
150  VDU 29,640;512;
151  REM ** MOVE GRAPHICS ORIGIN TO
     SCREEN CENTRE
160  FOR ANGLE=0 TO 84 STEP 6
170  GCOL 0,J
171  REM ** DRAW IN COLOUR J
180  J=J+1
190  MOVE Radius*COS(RAD(ANGLE)),
     Radius*SIN(RAD(ANGLE))
200  DRAW -Radius*COS(RAD(ANGLE)),-
     Radius*SIN(RAD(ANGLE))
210  MOVE -Radius*SIN(RAD(ANGLE)),
     Radius*COS(RAD(ANGLE))
220  DRAW Radius*SIN(RAD(ANGLE)),-
     Radius*COS(RAD(ANGLE))
230  NEXT ANGLE
240  REPEAT FOR I=1 TO 15
250  VDU 19,(I-2)MOD15+1,0,0;0
260  VDU 19,I,7,0;0
270  TIME=0:REPEAT UNTIL TIME 10
271  REM ** DELAY OF 0.1 SECONDS
280  NEXT I:UNTIL 0
281  REM ** LINE 240 ON ROTATES
     THE CROSS
```

PROGRAM LISTING 2

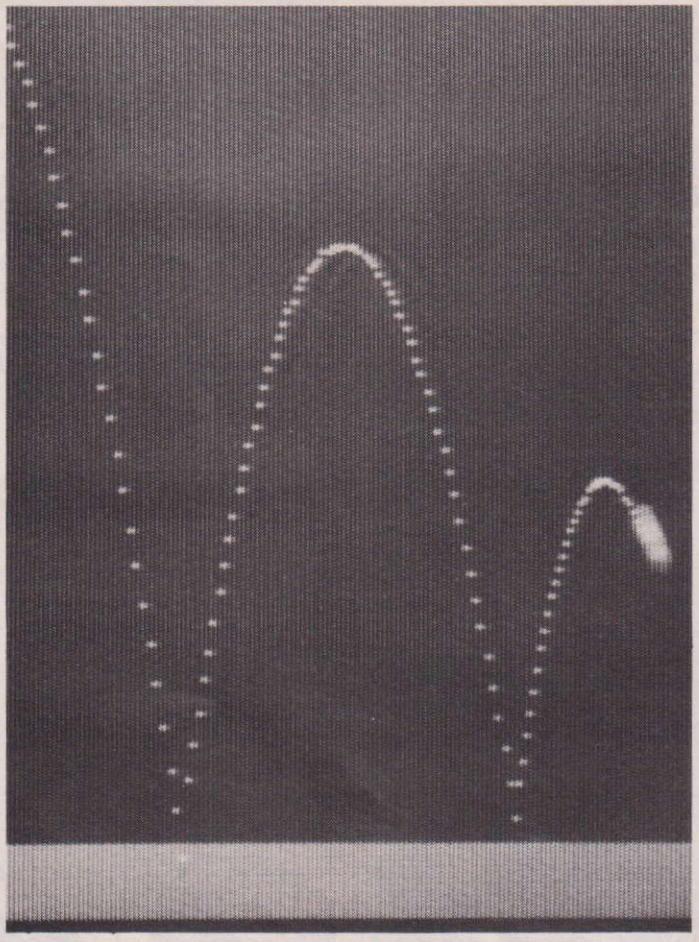
```
100  MODE 2:J=0
110  FOR THETA=0 TO 360 STEP 5.1
120  GCOL 0,J MOD 7+1
130  VDU 29,640+300*SIN(RAD(THETA)),
     512+300*COS(RAD(THETA))
140  MOVE 200,0
150  FOR ANGLE=0 TO 360 STEP 10
160  DRAW 200*COS(RAD(ANGLE)),
     200*SIN(RAD(ANGLE))
170  NEXT ANGLE
180  J=J+1
190  NEXT THETA
200  REPEAT FOR I=1 TO 7
210  VDU 19,I,(I-1+J)MOD 7+1,0;0
220  NEXT I
230  UNTIL 0
```

BBC Software Reviewed

A Gollner

THE COMPUTER PROGRAMME PROGRAMS — VOLUME 1

Check out whether the BBC's own software comes up to scratch in our complete software review.



OWL A simple but clear program which displays the symbol of the computer literacy project, an owl. It illustrates how easy it is to change the programs in this pack—with many clear statements which tell the programmer what is happening and how the programs work. It also shows how the computer creates characters. The picture it creates can be altered by changing clear and straightforward lines at the end of the program. However, it is a somewhat long program considering what it does.

BALL A program which shows how the computer can be used to simulate events that occur in real life. By changing one of the conditions concerned in bouncing a ball, for instance, we can predict how the ball would bounce. The display first shows a ball bouncing across the screen and leaving a trail of white dots. This procedure is repeated twice with different velocities and angles. Then, by pressing any key, the balls come bouncing back across the screen erasing the white dots. The action is accompanied by quite realistic sound-effects.

SALES This is a program which shows how the BBC Micro can display information on a graph. The program draws the graph axis and labels them, and asks you to put in your sales figures for each month. There are a few limitations, in that your sales must be between 0 and 900, and the programmer cannot specify what the units actually are (ie £, thousands, etc). But it does show that information, which cannot be extrapolated from a table of figures, can be displayed very clearly in graph form.

BUBBLE This program shows how the BBC Micro sorts data (in this case, words) into order, using very clear graphics. The words are displayed clearly and they light up when the computer swaps them into alphabetical order. The words which are put into order can be changed with ease. This program, with its clarity, looks similar to the program in the 'Welcome Pack' called 'Bubble-sort' and could have been expanded a little.

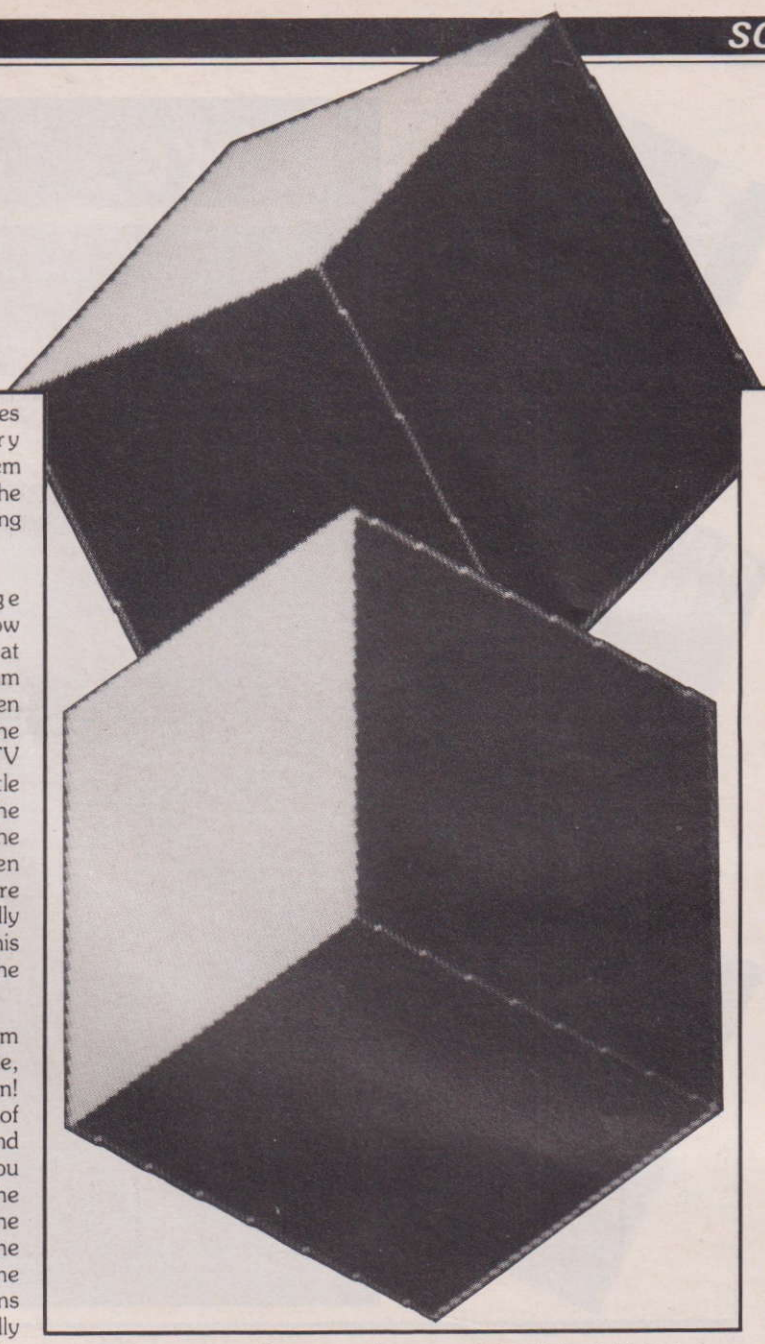
CUBE This is a program that shows how the computer can animate figures which are drawn on the screen. In this case, a simple cube is put into a spinning motion and after colouring in one side, the cube is revolved, using a colour-switching technique which is used in many of the programs to simulate animation. Could do with more colour and sound.

LANGUAGES A program that places names of various computer languages at random positions on the screen. A simple task you may think, but there is an interesting programming routine I can't fathom, which magnifies the letters to double their height. A nice effect but such a very long program!

FLOWERS Short and sweet. This program shows how much the plotting facility of the BBC Micro can do with two polar equations. A line is plotted from one side of the screen to the other with loops and twists, to the accompaniment of a string of notes from the sound command. A good economical program showing the graphics to good advantage in a clear, short and easy to read form.

ANAGRAMS A program which asks for a word and proceeds to print out all the combinations of it on the screen. The simple clear display helps, but there is a problem with the number of anagrams possible with the average word—there are too many! The words usually take up five or six screenfuls, scrolling when a key is pressed. The program shows another kind of sorting and also demonstrates the possibilities of print formats and display tactics, but you really need to go into the program yourself to find these pearls of wisdom.

WEAVING This is a program that shows how the computer can store such a large amount of numbers and pieces of information. It also shows the amount of combinations possible



with a few threads. It illustrates the principle of binary notation—the numbering system used by the computer. The graphics make this quite pleasing to the eye.

SIDEWAYS A strange program, which shows how versatile the computer is at altering the screen. The program first prints a letter on the screen (Dear Mr Bloggs, etc). But as the screen is shaped like any TV screen, the letter looks a little strange and squat—so the computer goes on to print the same letter on the screen sideways, so it looks more normal (but you have to actually turn your TV on its side!). This program is a little long for the result, but it is quite striking.

COINS A confusing program which asks you to solve a puzzle, before you turn the computer on! You have to get yourself a pile of coins (of which one is a dud) and a weighing machine, and you have to tell which of them is the dud by weight. You weigh up the coins and tell the computer the results of each weighing, and the computer tells you which coins to weigh next, to eventually deduct which is the dud. Why can't we get the computer to do all the tedious work of weighing for us to begin with?

PLOTTER A superb program, which uses the plotting facility of the computer to great effect by drawing three-dimensional graphs, demonstrating just how clear and beautiful computer graphs can be. The equations to produce the graphs are displayed at the top of the screen. But what is the basis for these graphs and how do they work? The equations and colours employed can be easily changed at the end of the program listing.

This pack seems to be another 'Welcome Pack'-type and is designed to show off the capabilities of the BBC Micro and teach you more about actual programming. I consider this pack good value at £10. It perhaps, could have done with more use of the sound command to display this facility, however, the documentation is very good, in that it has clear program notes and useful suggestions for improvement.

THE COMPUTER PROGRAMME PROGRAMS — VOLUME 2

GLOBE A program which displays a globe. It then shades it and prints lines of longitude and latitude, using a colour-switching technique to create movement by making the longitude lines run around the globe. I think that we could, perhaps, have done with a more addictive program at the beginning of the pack!

WATER & POND These are a pair of programs which were written to answer a particular problem stated on the BBC Computer Programme, but they show how the computer can answer questions in different ways. The question is 'Imagine we have an ornamental fish pond, in which swims a golden carp. It contains 500 gallons of water, but there's a leak and after each hour we lose 1/500th of the water in the pond. The carp will be in danger when there is less than 100 gallons left in the pond, so how long do we have to leave it before the carp is in danger?' Who knows? The computer does . . .

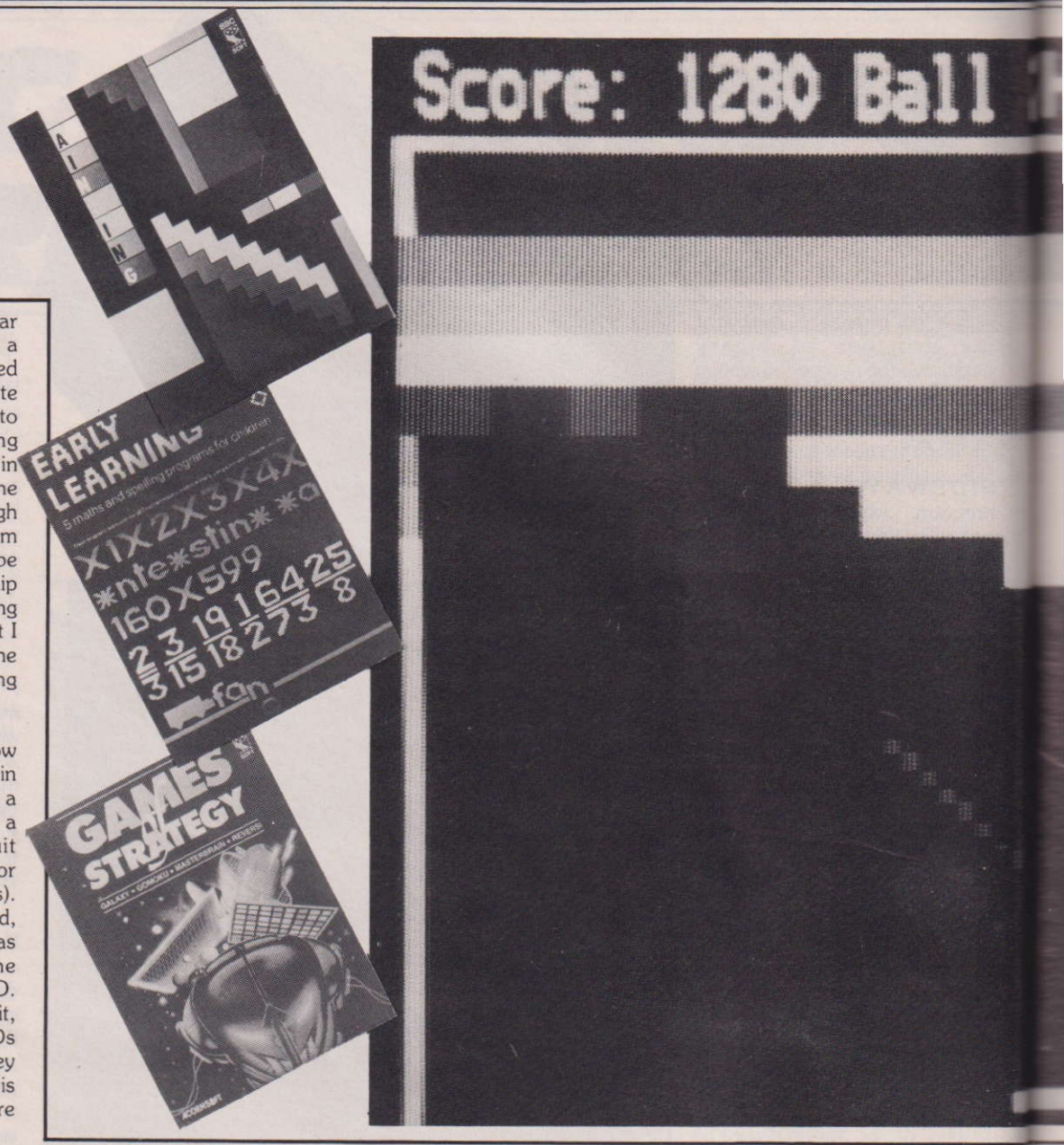
WATER This tackles the above problem by drawing a clear graph with a grid. In this way it is easy to find the right answer. The program illustrates how the computer can repeatedly do mathematical operations at considerable speed that would be too boring or difficult for us.

POND Solves the problem in a more interesting, if less accurate way. A cross-section of the pond is displayed showing the leak in the pond, a fish, reeds and a clock. When the water level reaches the 100 gallon mark on the side of the pond, the clock stops.

PARASER A strange title for an interesting program. It displays the basics of how the computer communicates with the outside world. It shows how a parallel interface works by printing vertical text through a set of parallel 'wires'. The same text is then printed through a single 'wire'. It shows how the speed of one compares with the economy of the other. This program could do with some sound, but the graphics are quite clear and achieve the aims set for the program.

LUNAR LANDER This is not only a game, but an example of computer simulation. A topnotch version of the old favourite. The program includes all the variables to do with the landing of a moon craft, as it gets closer to the moon's surface. As the gravity increases, the mass is reduced because the fuel is used up. Your controls are three rockets—up, left and right. However, this program differs

CONTINUED OVER



from the usual versions of 'Lunar Lander' in that as soon as a rocket is turned off, the speed still increases and the opposite rocket has to be activated to counteract your increasing speed. When the thrust is zero in both directions, the speed of the craft does not increase although it still moves from its momentum and the opposite rocket has to be re-activated to slow the ship down. A difficult but challenging game, which I must admit that I have failed to master as yet. The sound effects accompanying your moves are spot-on.

LED This is an example of how the BBC Micro can be used in education to present ideas in a clear graphic way; in this case, a design showing the circuit diagram of an LED (as used, for example, on calculator displays). As consecutive keys are pressed, a part of the LED is lit as well as the wires that carry the electricity to light up the LED. After all seven segments are lit, the program lights up the LEDs in the shape of the number-key you are pressing. Again there is no sound but the graphics are very good.

QUIZ The BBC Micro in education—a program to show how the computer can teach by asking questions with graphic display to keep the pupils interested. This program draws a single map of Britain and asks four questions about cities in Britain by marking the city chosen by a red spot indicating the position of the city. There is a great deal of room to put in your own questions and it is quite simple to change the display, but I would have liked to have seen some questions put in to begin with.

NOUGHTS AND CROSSES Sounds like a pretty simple program, but it has no strategy like its human opponents. As the games progress, it notes your strategy for later use and after about 50 games it becomes hard to beat. The board is very clear and the scores on the side are good—with clear numbers and precentages so you can see the computer's progression as it learns to beat you.

The COMPUTER PROGRAMME PROGRAMS VOL 2 is a good buy for model B owners who want a 'Welcome Pack' that has programs especially for their machines using all the advantages of the expanded memory. Though these programs lack sound (except for the Lunar Lander), they are a good set of eight programs and well worth £10.

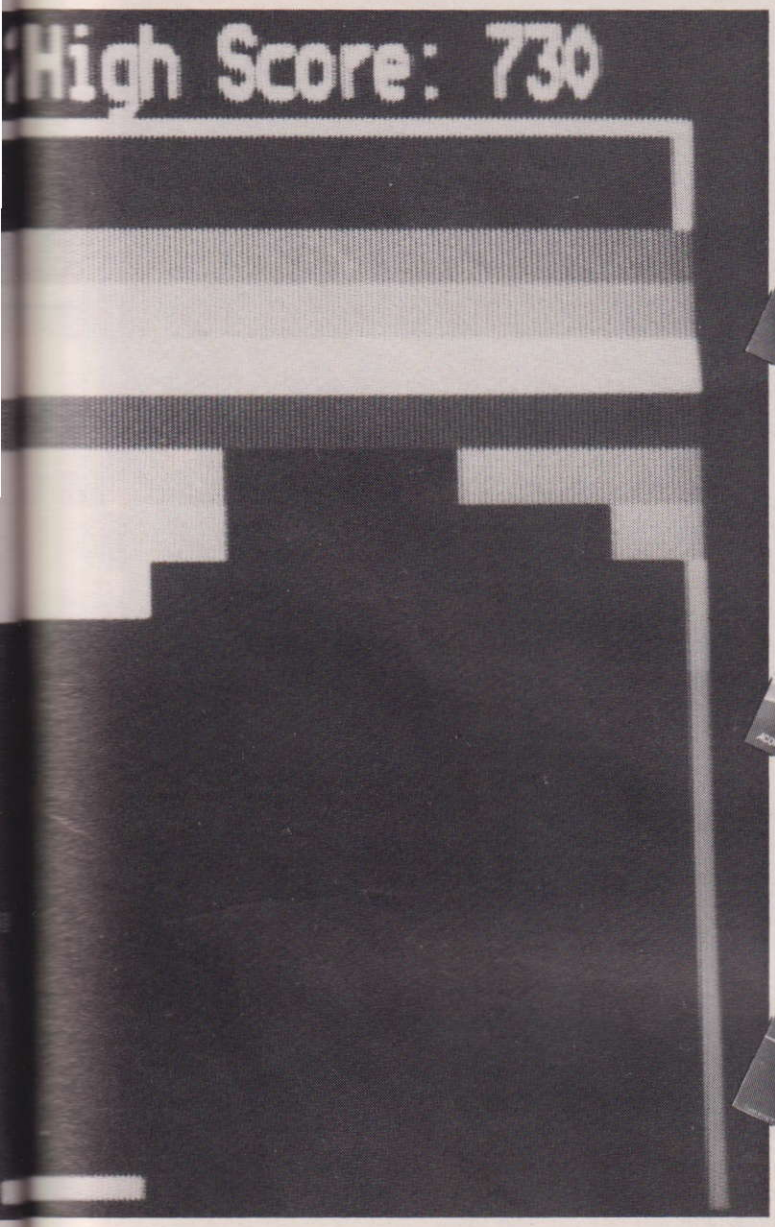
CLARITY & ADAPTABILITY	50%
SOUNDS	15%
GRAPHICS	60%
DOCUMENTATION	60%
VALUE FOR MONEY	55%
OVERALL VERDICT	65%

%'s for previous pack

CLARITY & ADAPTABILITY	50%
SOUND	70%
GRAPHICS	60%
DOCUMENTATION	70%
VALUE FOR MONEY	70%
OVERALL VERDICT	70%

ARCADE ACTION

BREAKOUT The aim of this game is to knock down a wall of coloured bricks using a ball, controlled by a bat. You are allowed three balls—if you miss a ball with the bat, you lose it. As the ball hits the bat, the sides of the screen or the bricks, the computer bleeps—the further up the bricks are, the higher the note, and the higher your score. The game is challenging and difficult. The further up you smash through the wall, the faster the ball gets and the games becomes progressively harder. The game has brightly coloured brick sections and the bat can move fast. The ball moves smoothly and bounces well. The score is very clear at the top of the screen. You can choose whether you want to use a large or a small bat. The high-score 'ovation' you get at the end is well worth playing for.



CLARITY & ADAPTABILITY	65%
SOUNDS	60%
GRAPHICS	65%
DOCUMENTATION	70%
VALUE FOR MONEY	70%
OVERALL VERDICT	65%

GAMES OF STRATEGY

GALAXY This just qualifies as a game of strategy, but is very good with semi-real action. What you have to do is to destroy the alien race known as the Kyrons. The way you go about this is to use your ship 'Endeavour' to go to various sectors of the galaxy to destroy the Kyrons, before they attack you. The Teletext display is simple but effective. You can choose between three levels of difficulty and your aim is to destroy about 50 Kyron ships in a specific time. There is quite a lot of position play but there are a deal of ship encounters. There are three possible views of where you are, the local sector, eight parsecs by eight parsecs in which stars, bases (where energy is recouped) are shown. Then, there is a long range scan showing the surrounding eight sector and finally, there is the galactic situation based on information that you have discovered situation based on information that you have discovered through the long range scan. This galaxy you have to clean up is eight sectors by eight sectors. The dangers of this game are running out of energy or being destroyed by the Kyrons. Their weapon is the phaser—a kind of energy weapon which irradiates in all directions and depletes your shield energy—which you set through one of the nine controls. '1' moves you about; '2' gives a long range scan; '3' Phaser energy (depletes their shields); '4' Photon torpedos—you have

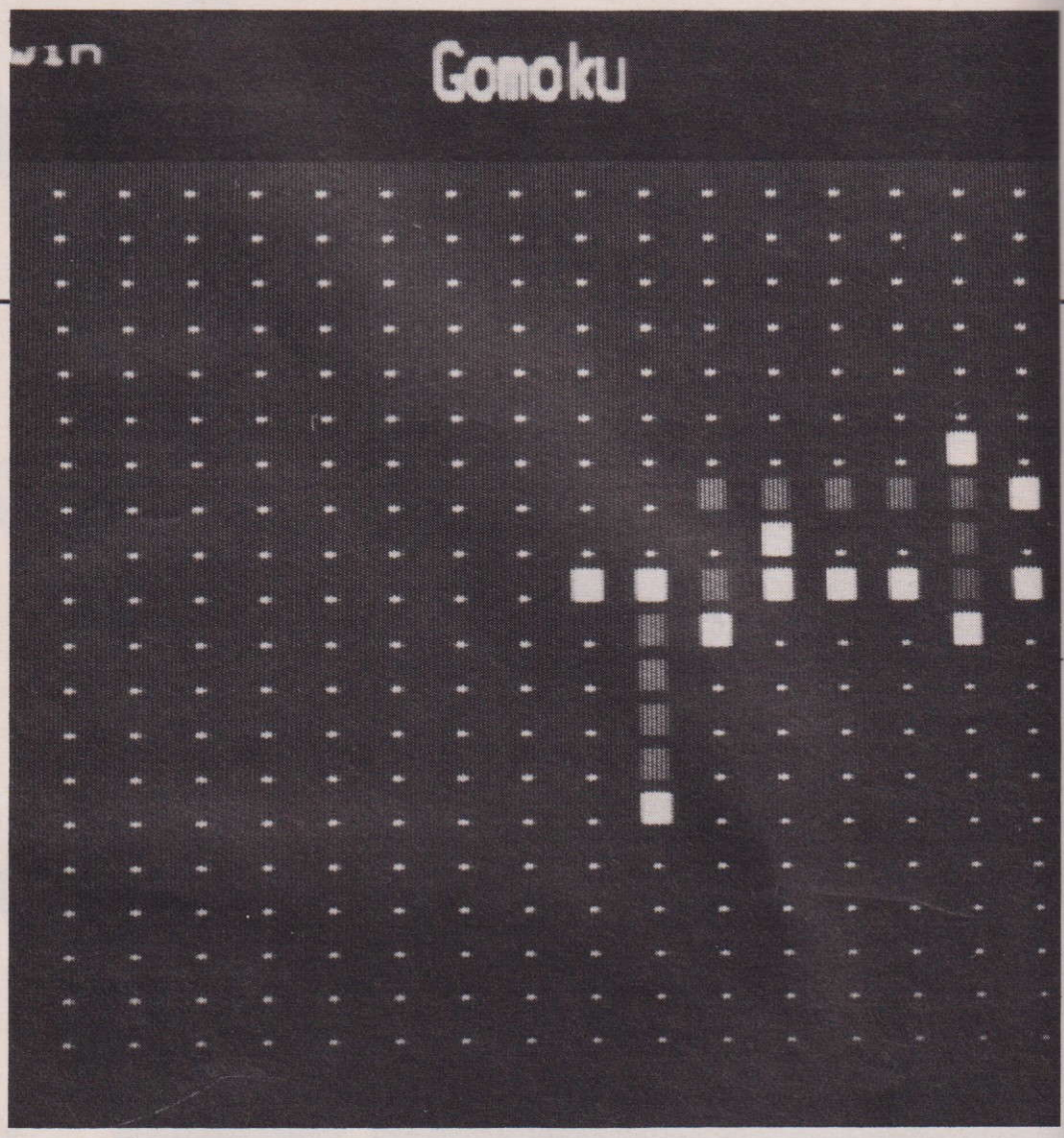
CONTINUED OVER

DODGEMS This is a program which is very similar to a popular arcade game in which you pilot a small racing-car around a track eating up power pills, but you have to watch out for another car controlled by the computer which tries to stop you by crashing into you. I found the controls a little difficult to manipulate but I soon got the hang of it. The racing-cars look quite realistic—but there weren't many colours considering it can be done in Mode 2 for Model B users. The idea of the game is quite good, but the score is a little basic in that there is an incrementing number in the centre but no high score. There are no second chances—you get one car and there are no faster or slower cars to choose from.

FLASH is a 'Simon'-type game in which you have to repeat a series of coloured patterns on the screen with the help of comments in the centre. Also, musical notes are sounded corresponding to the colours and positions on the screen. This is a rather familiar game, but this version is bright and clear with its simple squares filling up with strong colours. You get a score result right at the end.

SNAKE is a fast-action graphics game in which you pilot your snake across a screen to where it can find letters of the alphabet which it eats. The letters appear randomly on the screen and as the snake eats them it grows by the 'value' of the letter (A=1, B=2, etc) and becomes more difficult to manoeuvre, because you are not allowed to run into your own snake's trail. Your score, plus the previous best score, are displayed at the top of the screen. The snake moves very smoothly and is a joy to watch. The colours are clear and there is a good gulping sound when the snake swallows a letter and a raspberry when the snake 'crashes' into itself, or crashes into one of the walls. If you take your time in getting a letter, its value quickly reduces and it eventually disappears.

FUNGAMES is very good value at £10 for four great games which take advantage of the speed of BBC BASIC and the sound and graphics of the machine.



10—which are directional and destroy whatever is in their path; '5' sets your shield level to protect you from phaser energy; '6' gives you short range scan; '7' Galactic situation; '8' freezes the game; and '9' lets you relinquish command. Apart from the lack of sound, this is a very good and engaging game.

GOMOKU This is a very interesting game which contrasts with GALAXY in that it is very simple in operation but complicated in strategy. This ancient game could be called 'Mega noughts and crosses'. The differences are that the grid is 19 by 10 not three by three, and the aim is to get five in a row of your counters. You simply move your cursor with the cursor keys and press Return when you want to place your counter and see where the computer wishes to put its counter. A nice touch is that the computer's counter flashes a while so you can see where it went. The computer is a good opponent and the clear clean graphics are good, but again there is a lack of sound in the program apart from the ovation at the end. This is a game well transferred from board to computer and at the average price of £2.50 in real terms, it is cheaper than the board game in the shops.

MASTERBRAIN The computer version of a game which has brought a deal of money to people who revived the ancient game of 'Bulls and Cows'. In this game your opponent thinks of a four figure number and once you have made a guess at it, they tell you how many figures you got in the right place and how many in the wrong place, and you have to deduct which numbers these are, and thus the number he or she is thinking of. A simple game which has been given a little twist in this program in that you guess the computer's number while it guesses yours. The game is also lightened by coloured numbers to help you if you want to play with colours. The display is quite good but looks a little cramped. The colours and numbers could have been accompanied by sound for more recognition.

REVERSI This is a game also as ancient as 'GOMOKU' and 'MASTERBRAIN'. The idea is to capture the most counters on the eight by eight grid. The way to capture counters of the opponent is to have your counter one side of the opponent and then one on the other, and as you capture a counter, it is turned over and becomes one of yours. The game is complicated, and the winner is the one with more than 32 counters at the end. There is a '?' message for you to decide whether the computer plays against itself, against you, or you can play against your own opponent.

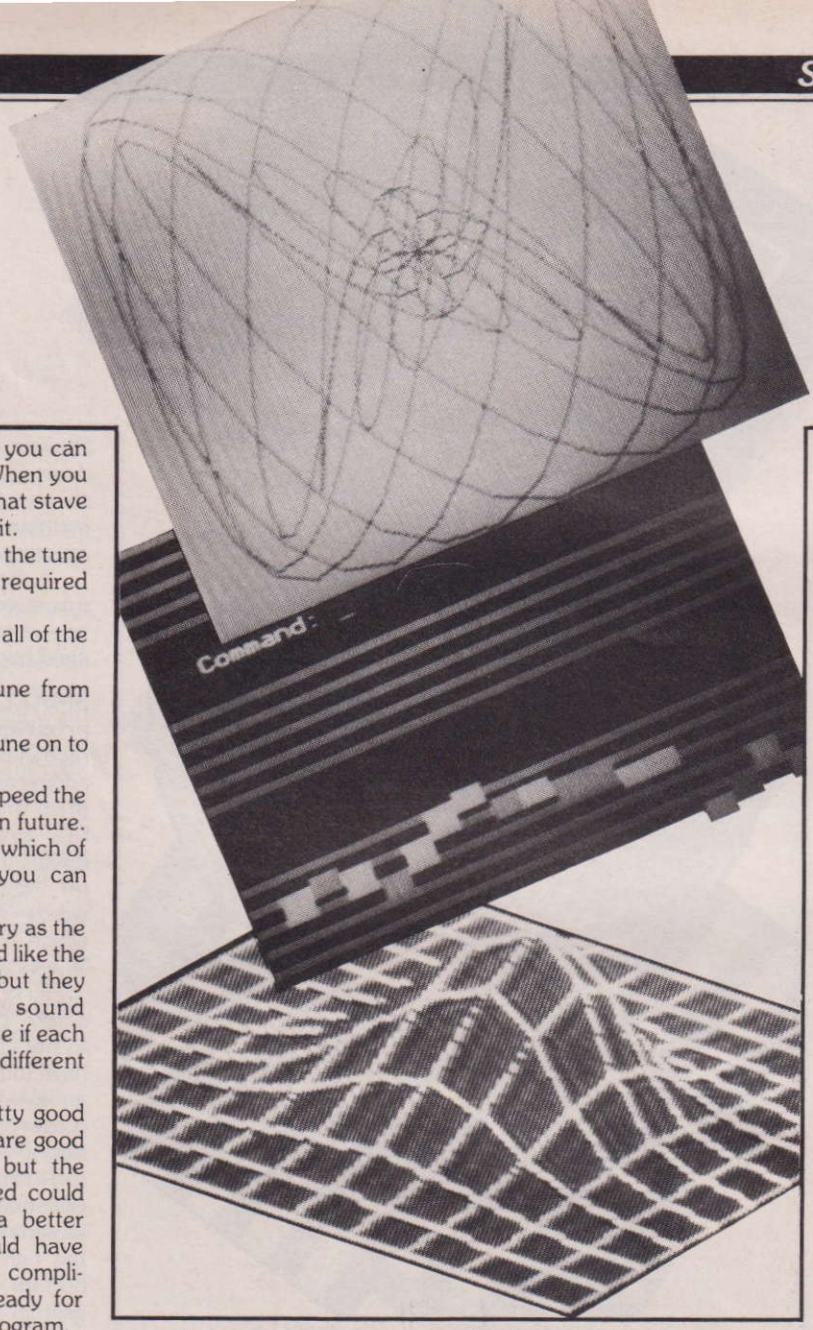
All in all there is very good value in this pack since to buy three of the games in the shops would add up to about £15 at least. Also, these programs are not only great for accessibility and easy play but good on strategy. Like the tagline for REVERSI, 'A minute to learn, a lifetime to master'—that is what games of strategy are about so the inclusion of GALAXY is a little unadvised, even though it is a very good program. It is a pity that sound is not employed apart from the winning ovation at the end. But all in all, a pack that is well worth buying.

CLARITY & ADAPTABILTIY	50%
SOUNDS	10%
GRAPHICS	50%
DOCUMENTATION	60%
VALUE FOR MONEY	70%
OVERALL VERDICT	65%

MUSIC

This is a pack which enables you to write your own tunes on the screen and then play them back using staves and four different instruments. You input your tune using 22 keys—a range of nearly two octaves. Apart from these keys there are seven controls as well as the cursor movement keys for left and right and delete. The seven controls are on the red function keys.

f0 is the Record/Edit mode in which you put your tune into the computer using the Delete to rub out notes and the note keys. The Space Bar can be used for rests. You can also use the cursor keys to place new sections of notes into a tune at any point, and as you move the cursor over a note,



it is played for you and you can delete that if you like. When you press this key it asks what stave you want to Record/Edit.
f is the key which plays the tune either in one of the required staves or all of them.
f2 lets you delete one or all of the staves.
f3 lets you LOAD a tune from tape.
f4 lets you SAVE your tune on to tape.
f5 lets you decide what speed the tune is to be played at in future.
f6 enables you to decide which of the four instruments you can play with.
These are pretty arbitrary as the notes do not really sound like the instruments indicated, but they do make the tune sound different. It would be nice if each stave could have a different instrument.

The program is pretty good itself in that that tunes are good when you play them, but the notes as they are played could look better by using a better mode. There also could have been some sufficiently complicated tunes on tape ready for input to illustrate the program.

CLARITY & ADAPTABILITY	60%
SOUNDS	50%
GRAPHICS	50%
DOCUMENTATION	60%
VALUE FOR MONEY	55%
OVERALL VERDICT	60%

DRAWING

This pack lets you draw complicated things on the screen with a range of effects which can be assessed from the 10 function keys. You draw by selecting a direction using the cursor keys or a combination of them, then select a velocity using the number keys from 0-9. Here are what the functions can do:

f0 clears the screen.
f1 lets you change the foreground and background colours to any of the 16 available.
f2 lets you define the width of your nib.
f3 is a very good effect which makes polygons of any number of sides from triangles to circles. There are also two radii involved, so you can create eclipses and rectangles and really any shape you want. Also, you can use a polygon as a nib by moving the cursor and pressing the Space Bar to repeat the figure.
f4 is similar to f3 but it draws spheres with lines of latitudes and longitude with any radii, so you can get an oblong spheroid if you like.
f5 is exactly the same as f3 in that you are asked to give information regarding to a polygon, but the difference is that you move the cursor to another point away from the centre of the polygon and it will draw lines from that point to the vertexes of the polygon to create a prism or cone in the case of a circle.
f6 lets you create any mesh at any part of the screen.
f7 gives you a set of preset horizon lines on the screen.
f8 lets you use some text on the screen either to table things or to use the letters as a kind of strange shaped nib.
f9 enables you to stop drawing.

All in all this is a program which has many advantages and helps you to design complicated and beautiful patterns. However, some of the GCOC functions could have been employed to greater effect.

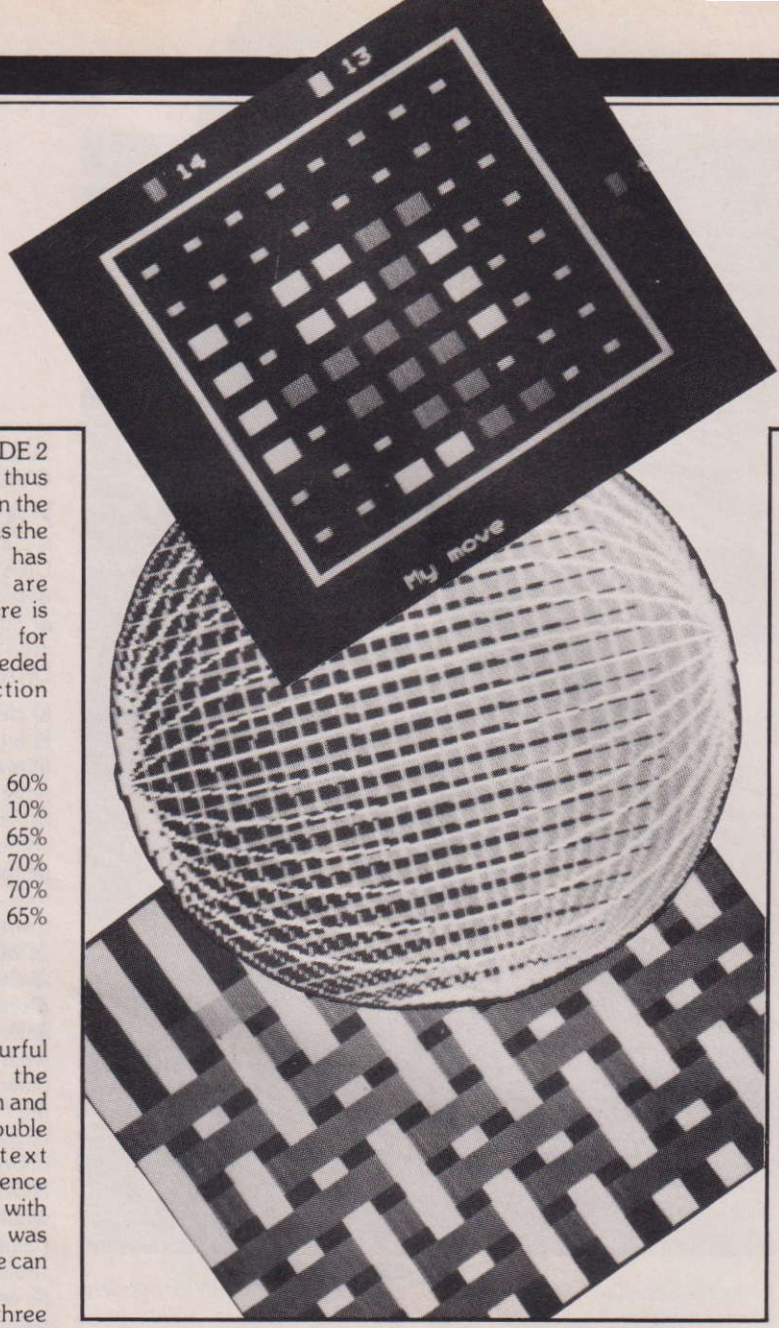
CLARITY & ADAPTABILITY	60%
SOUNDS	10%
GRAPHICS	65%
DOCUMENTATION	55%
xxxxxx	
DOCUMENTATION	65%
VALUE FOR MONEY	55%
OVERALL VERDICT	50%

PAINTING

is a similar pack to the DRAWING pack but obviously is based on the drawing and mixing of colours on the screen. There are many functions, like the DRAWING pack.

f0 clears the screen.
f1 lets you choose which of the four available colours you wish to paint.
f2 lets you change the actual colours that are available for painting, but you can only have four at the same time to choose from (but these can be any of the 16 available).
f3 decides how the paint will go on the screen and OR makes dark colours go lighter colours when they are painted. XOR makes a new colour appear when the two colours are painted together and gives the impression of the colours mixing. NORMALISE is the way you would expect things to be painted—the first things under, the last things over.
f4 enables you to start actually painting with an implement, which is this case is an airbrush which can be any width. As you move the cursor around, a series of random dots appear around your brush. Sounds weak, but it is effective.
f5 if a function which lets you pick up a paint brush which can be any reasonable size and either the shape of a triangle or a square which you can paint with.
f6 is the shading function, with which you give a width and a scale factor, and a square of shading appears by your cursor.
f7 is similar to the polygon function on the *drawing* pack except the polygons are filled.
f8 is for text to be used as a shape for painting—so letters can be bounded across the screen.
f9 lets you stop painting.
This program could have been

CONTINUED OVER



adjusted so it goes into MODE 2 for Model B owners and thus create unexpected colours in the XOR painting mode. This has the functions of drawing but has good functions which are applicable to painting. There is again no sound except for prompting, but none is needed really apart from function recognition by note.

CLARITY & ADAPTABILITY	60%
SOUNDS	10%
GRAPHICS	65%
DOCUMENTATION	70%
VALUE FOR MONEY	70%
OVERALL VERDICT	65%

EARLY LEARNING

FRAC This is a colourful program which teaches the basics of fraction conversion and fraction addition. The double height colourful Teletext graphics make a great difference to the display and help with readability. The program was divided into sections that one can progress through.

The first shows you three kinds of fractions in the top part of the screen, while on the bottom you table fractions the computer gives you.

The next part is concerned with changing 'top heavy fractions' into mixed numbers, and the display is similar to the previous section. When you take too long to answer a question or get it wrong, the computer goes to a 'help' page which clearly and succinctly explains in detail how to carry out the operation.

The third part works in the same way but is concerned with the translation of mixed numbers into top heavy fractions.

The fourth and fifth parts are concered with equivalent fractions and fraction addition with no examples, but there is a 'help' page.

Overall, a good, clear, if slightly cold, way of teaching fractions.

MULTIPLICATION This is a program which shows how to do long multiplications. There are two parts—questions which the computer goes through step by step, then there is a test section which tests to see what you have learned.

The first step by step conducted tour through long multiplication is handled very well because in the program, the number that has to be individual sums that make up long multiplication are printed on the right. The computer shows how much it is carrying when right. The computer shows how much it is carrying to when multiplying and when adding.

This and the other section of this program are very clear and easy to understand and would be good for those who found long multiplication a hard part of arithmetic.

TABLE This is a simpler multiplication program than the previous one, which concerns the recall of boring multiplication tables. There are, again, two parts.

The first asks you which of the 12 times tables you want to be tested on—and proceeds to test you on the table in order 1×7=?, 2×7=?, etc. then when you have completed that, the computer picks out random questions from the table for you to answer and then goes back to the menu.

Then, there is the test section which consists of 20 questions which you have to answer in 10 seconds. When you have finished these, you are marked down the screen with ticks and crosses and then you are tested on them using the first part of the program.

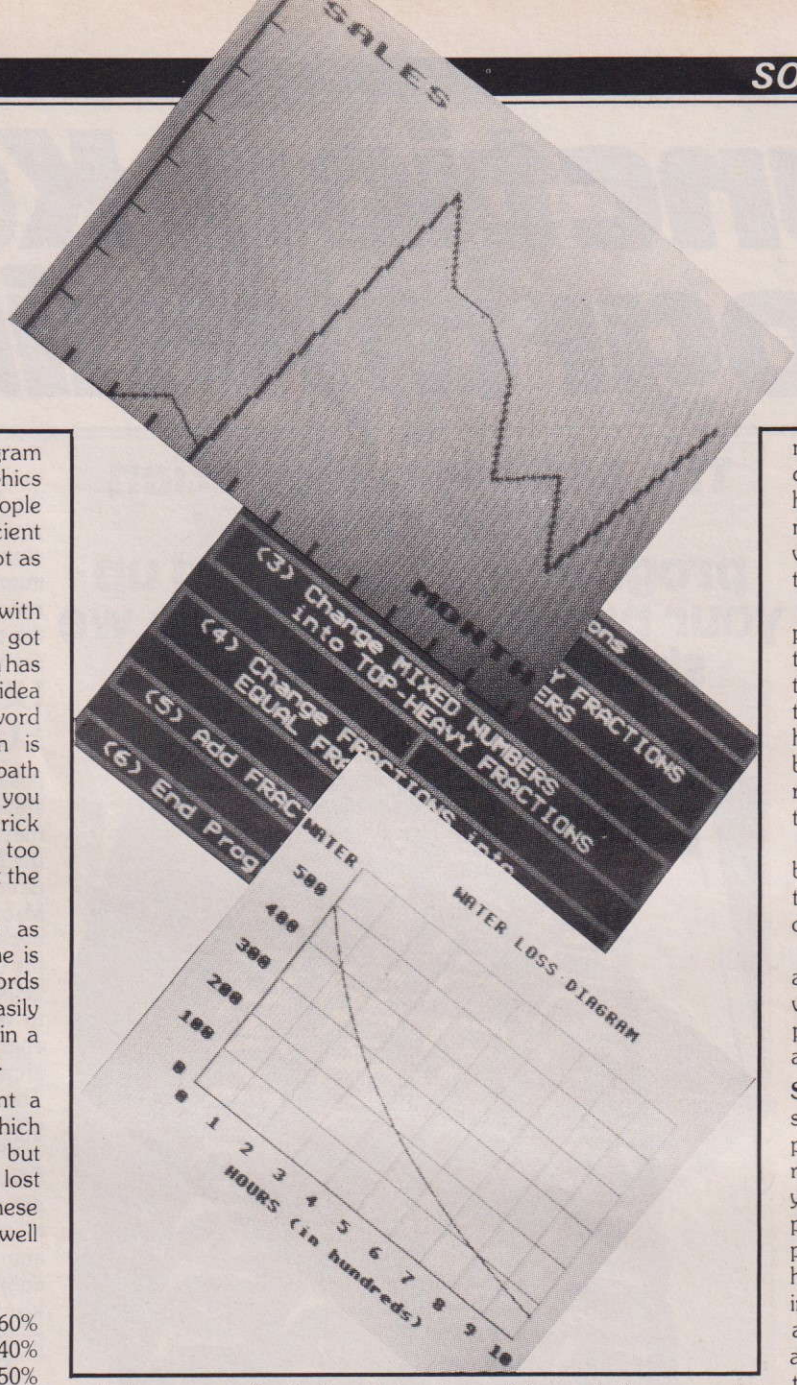
A program which seems a bit questions—like the other programs it would be nice if the child had a goal.

SMALLDOZER This is a program which tests the recognition of words and how adding an 'e' on the end changes them into gibberish or changes the pronunciation and the meaning. This program is based on how well you can find words—from four on the screen which would still make sense if an 'e' was added on the end.

First, Billy Bulldozer whirrs across the screen and then we go to the first page. On the left, we have four three letter words with an 'e' a little to the right of them underlined and in double height—with the vowels in a different colour from the consonents. There are also underlinings with no words on the right of the screen. At the bottom of the screen there is a prompt to put your own word in which is a word made up of one of the four three letter words with an 'e' on the end.

Billy Bulldozer then appears on the screen by the group you chose and he pushes the letters towards the letter on the end of the underlining. Then it pauses so you can look at the new word. If the new word makes sense, a bridge appears between the left underlining and the right and the 'dozer pushes the word across to the other side. If, however, it is not a proper word and does not make sense a 'bridge' does not appear and the letters tumble down into the chasm.

This is a very good program but it seems to be a little too specific, and could be widened to spelling—like having anagrams and having to unravel them properly otherwise the 'bridge' may not appear.



FUNNYMAN This program is a good way of using graphics and an old game to teach people to spell. It is based on the ancient game of Hangman, but is not as violent.

Jumbo wants to be fed with some water and Jim has got it—but the path to the trough has a few bricks missing and the idea is to guess the letters in a word and for every letter which is right, a brick appears on the path to Jumbo's trough. But if you guess a wrong letter, a brick appears in a wall and if it gets too high Jim won't be able to get the water into Jumbo's trough.

The change of scene as regards the plot of the game is good and engaging. The words that are guessed can be easily changed and are displayed in a colourful form—in MODE 5.

EARLY LEARNING present a good set of programs which could teach a great deal, but some marks have to be lost of the programs. But these programs are clear and well suited to their purpose.

CLARITY & ADAPTABILITY	60%
SOUNDS	40%
GRAPHICS	50%
EDUCATIONAL VALUE	70%
DOCUMENTATION	70%
VALUE FOR MONEY	65%
OVERALL VERDICT	65%

FAMILY FINANCE

HEATING COSTS The idea of the program is to input your current heating facts and figures and see how much you are paying, and then to let you change some of the variables—use a different fuel, get double glazing, insulation in the loft, lower the average temperature in the house—and so on.

The final printout you get is very good and useful but there is a lot of information that you have to find out and put in to get realistic answers. So the booklet you get with you packs tells you all the things you will be asked and leaves spaces for you to fill in your particulars for use when you run the program.

The visual side of asking the question is a little stark—white writing on a black background, scrolled. A little graphics would not waste memory but would liven up the program a little. But this program does what it sets out to do clearly and concisely.

RENT OR BUY A very good program that has the advantages of clarity of the previous program but uses graphics to better effect. The program was both useful and interesting—a rare quality in programs that I have seen. You input some information about the article, and the computer shows, using a series of clear tables, the best way of owning the article, by renting or buying.

The initial price, the rental instalments, costs and maintenance contracts are taken into account, even the money you are losing by not putting money into a building society is shown on the tables. These tables show money facts for each of the five years that you are renting or buying, so the facts can be gleaned from columns on the table. A very good program.

BORROWING This is a more basic program which calculates the costs of borrowing either for a mortgage, hire purchase or just a lump sum. The program asks you your tax rate and then when more information is added, displays a table showing details of how much interest there is, how much cash extra do you pay, what the real percentage is after tax relief and so on.

The main fault with this program is that it assumes that all the interest rates are the same as they were at the beginning of this tax year, so you input your highest rate of income tax at the beginning and it displays the rates that it will use in the rest of the program.

You can change these rates by getting into the program but that is a bother and the changes could be made by the program.

This is a good program to run and it does help give the facts when all you have got as a vague percentage and a timespan from a dealer.

SAVING This program is similar to the BORROWING program in that it has assumed rates of interest, but in this case, you can change them in the program. This means that the program is quite simple in that it has a few tables giving information on percentage rates and using a tax percentage, and an inflation rate that you put in to adjust the rates to your situation. Since there are a large amount of things to be changed, you can save the program onto another tape. More of a database than program, but useful nonetheless.

FAMILY FINANCE is a good value pack which does exactly what it sets out to do in a clear fashion. These programs are a boon to family finance and would do well to be at most Citizen's Advice Bureaux if not on each BBC Micro.

CLARITY & ADAPABILITY	50%
SOUNDS	10%
GRAPHICS	40%
DOCUMENTATION	70%
VALUE FOR MONEY	70%
OVERALL VERDICT	65%

Function Key Programming

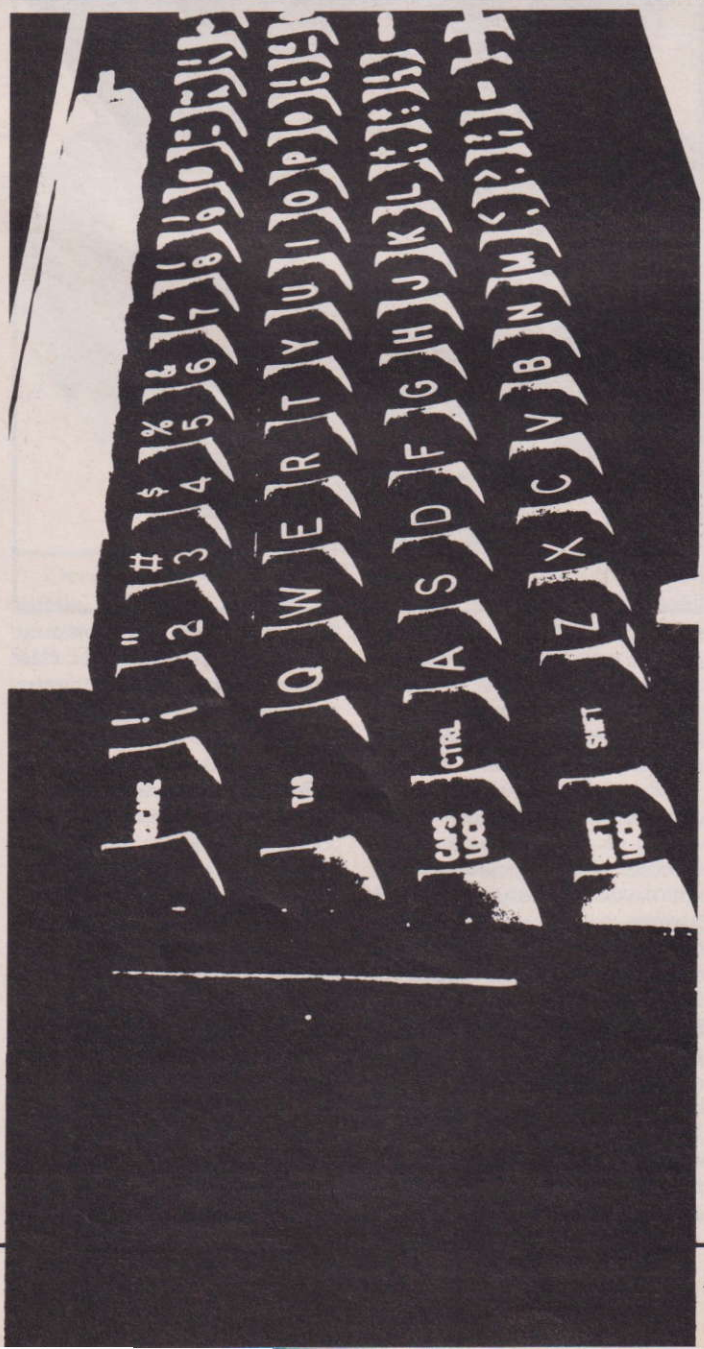
The BBC Micro's function keys can be easily programmed to speed up your programming. Here we show you how to do it quickly.

Your BBC Micro has actually arrived. The novelty has had time to wear off and you are beginning to write your own programs, instead of playing 'bat'n'ball' or running your biorhythms. You have connected your Model B to a printer and your study is full of old listings, none of which being recognizable as the current listing.

This article is designed to help you by printing the date and time of your old listings, as well as helping you to control some features of the BBC Micro that are a mixed blessing. For instance, variable names in lower case are easily distinguishable at a cursory glance from upper case BASIC reserved words and, more important, you can actually use the reserved words themselves in lower case variable names. How often, though, do you then type run or list exactly like that, in lower case, and get the dreaded message 'Mistake'. Of course, it's a mistake you mutter, as you reach for the Caps Lock key, you wouldn't be fool enough to do it deliberately!

Or how often do you press Break to get your program out of a mess without leaving yourself in graphics mode, with enormous pulsating letters in flashing green-magenta on a flashing red-cyan background (try listing your program that way to find errors . . .). Having pressed Break and forgotten to type OLD (such a useful feature in itself but easy to overlook) you type in a PRINT statement to debug the latest error and lose all the amendments you have laboriously traced and entered in your program—and that always happens just before you save the definitive version of a new masterpiece.

Again, if you have a Model B driving a printer, you may have become accustomed to preventing scrolling of your listings by CTRL/N (cancel it by CTRL/O and don't take any notice of the provisional manual on page 107). It is not uncommon to direct a listing to the printer by the handy CTRL/B, to find that you not only have CTRL/N stopping the listing at frequent intervals, but that long lines are running off the right-hand edge of the paper, because you forgot to set WIDTH 80 or whatever.



USER-DEFINED FUNCTION KEYS

It is to other pioneers of the art of staying sane long enough to master interactive graphics on the BBC Micro, that this article is directed. The top row of the BBC Micro keyboard, holds a decorative row of orange keys, the user-defined function or 'soft' keys, whose full capabilities have to be exploited by anyone who enjoys doing more than playing arcade games on their micro. Their speciality is this—that if one or more statements or commands is 'imprinted' on a function key by the user, then by pressing that key the user can invoke the imprinted sequence at any time when the micro is in command mode or waiting for keyboard input during a program.

First thoughts on putting the function keys to work during a program development session (you may work out for yourself how to use them to best advantage for program responses) lead to some very simple key sequences. Obvious sequences, each on its own key, are the most frequently used commands (in upper-case to prevent the confusion described above) namely RUN, LIST and OLD. You could put AUTO and RENUMBER as well providing you are that disciplined in your program development. However, we'll stick to the first three in this description, with one very special key which I shall come to a little later.

Using the BBC's function keys you can make your micro do more

G Pettit

LOADING THE SOFT KEYS

RUN is the easiest command to deal with. If you want to put it on Key 0, say, the correct way to program the key is with the command:

```
*KEYØ RUN¦M
```

where the character ' ' (shown as '¶' in Teletext) will be interpreted as CTRL. Those of you who are familiar with the ASCII table will know that Carriage Return in the ASCII table has decimal value 13 (&OD Hex). The value can be transmitted by sending the character CTRL/M (M being the thirteenth letter of the alphabet), and M on the function key is therefore interpreted as Carriage Return or CR. When Key 0 is pressed, the contents of the key will be displayed on the screen and the M executes the command, just as though you had typed RUN and pressed Return.

LIST is a more interesting command to put on a key. You may not always want a complete listing of your program, in which case you type, say, LIST 1000, 1200 to list all program lines between and including 1000 and 1200. You could therefore program Key 1 to display the command LIST but not to execute it (leave off the M) so that you can optionally enter line numbers before manually pressing Return. The key-loading command for this is:

```
*KEY1 LIST
```

If you thought we were going to put OLD on Key 2, you have missed an interesting feature. The position of the Break key on the keyboard, next to function key 9, is no accident. The Break key is, in fact, function key 10 and can itself be programmed. It will always re-enter the BASIC interpreter before displaying the commands programmed onto it, but this key is the obvious one to hold OLD for us, as we then get an automatic Break OLD sequence. In the rare case when we do want to delete the current program from memory, we can follow the Break OLD key by typing NEW, a safer state of affairs than having to remember to type OLD! The command to place OLD on the Break key is therefore:

```
*KEY1Ø OLD¦M
```

and you may, of course, use spaces to make the key-loading commands more legible, but the space available to all the function keys is limited to &100 Hex. Unnecessary spaces may therefore result in a premature 'Bad key' message if you are exploiting the keys to the full. By the way, before you write in to say the quotes have been left off these key-loading commands—the quotes are optional (unless you want to force leading spaces, which otherwisde are stripped) and so I leave them off to save typing time and error. The commands:

```
*KEY4REN.1ØØ,1Ø¦M
```

and:

```
*KEY4 "RENUMBER  1ØØ,1Ø¦M"
```

are identical in effect but the first is more economical of memory and quicker to type.

DOUBLE OR SINGLE QUOTES

The question of quotes in key-loading is an interesting digression. If the first character after the key number (apart from spaces) is a quote, then the whole key-loading expression must end with a quote. If you need a quote as part of the expression, it can be forced by two successive quotes; however, if quotes are not used to enclose the expression (ie the first non-space following the key number is anything other than a quote) then a quote is accepted as part of the expression.
For example, these are valid key-loading expressions:

```
*KEY5 SAVE "FILENAME"
*KEY5 "SAVE ""FILENAME"
```

but this would get a bad key message:

```
*KEY5 "SAVE "FILENAME"
```

and this would get a bad string message:

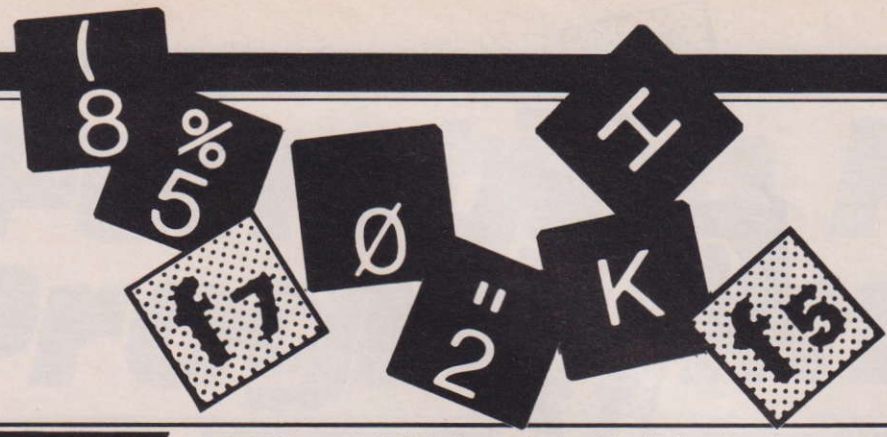
```
*KEY5 "SAVE ""FILENAME""
```

SOFT KEYS THAT WORK HARD

The program shown in Listing 1 illustrates these uses of the function keys, together with one rather more complicated use which will allow you to reproduce the date and time of day on your listings. This can be useful if you pick up an old listing in place of the current one. The time of day may seem frivolous, but if you spend a whole day working on a development program, it can be just as useful to know at which end of the day the listing was made.

You will already be aware of

CONTINUED OVER



the value of the TIME function, either in controlling delay loops or in monitoring response time; for example, in CAI (Computer Assisted Instruction) programs, where the pupils' performance can be judged on speed as well as on accuracy. You may not be aware that, although the **User Guide** warns that you can't use TIME to check the Greenwich Time Signal, it only loses six seconds in two and a half hours. Before the advent of quartz digital watches, that was par for the course! You can either prime the TIME function with the actual time of day, or by setting it to zero, you can use subsequent readings from it in 'stopwatch' mode.

THE SOFT KEY BUFFER

There is an interesting problem in forcing a date entered in response to an INPUT statement onto a user-defined key. Once the key-loading program shown in Listing 1 has been run, it is going to be overwritten by the programs you are in the process of developing or testing. The variables associated with this program, including D$, will be cleared as soon as NEW or LOAD is typed. Note that the soft keys are not cleared by NEW, LOAD, Break or by anything else I have tried them out on, except by switching off. D$ therefore has to be transferred to the part of the soft key buffer which holds the contents of Key 9, and line 310 includes a dummy message ('Date not known') preceded by an '*' to locate this dummy message area, even if line 310 is subsequently altered by users of the key-loading program.

In case the same idea is useful to readers for other key uses, I will show how the input date is transferred to the soft key buffer. This area extends from &B000 Hex and &C00 Hex and begins with 16 pointers, one for each of the keys 0 to 9, then one for the Break key 10, and five spare which don't appear to belong to any physical key. These pointers are offsets which, when added to &B00 Hex, point to the end of the previous key message. Any empty keys point to the last programmed byte in the soft key buffer and, as each key message is added to the buffer, the remaining empty pointers are pushed to the end of the new message. As soon as the buffer is full, the message 'Bad key' is displayed and any superfluous or extra long keys must be cleared (eg by *KEY 9) to release room for the new key.

Once the '*' in Key 9 has been located (lines 330 and 360) the first 14 characters from D$ are copied, or spaces are inserted if D$ holds less than 14 characters. The soft key buffer search and the transfer from D$ are done by '?' which in lines 330 and 360 acts as a PEEK and in line 390 acts as a POKE. The program can now be deleted since D$ is no longer necessary to hold the date; subsequently, pressing Key 9 will clear the screen and display the date and current time. If you wish to add them as a REM to the program you are developing, type a line number and the word REM, then use the Copy key to bring the date and time down into the new line. If you intend to have them printed at the head of a listing, then remember to press Key 7 first—but be prepared for the paper to shoot to a new page when the CLS is actioned!

This does make neat formal listings and, with any luck, you will never work from an out-of-date listing (or type RUN) again.

PROGRAM LISTING

```
  40 REM * Uses func. keys for calendar etc. - KEY 9 is left loaded for
display of date and time; use KEY 7 first to print them before listings
  50 REM
  60 REM  ***   Main program here
  70 CLS
  80 INPUT "Date (max. 14 chars.) - "D$
  90 REM * Any separator may be used between hh and mm
 100 INPUT "Time (24 hr clock) as hh:mm - "T$
 110 IF LEN(T$)<>5 OR VAL(MID$(T$,3,1))>0 THEN 130
 120 IF LEFT$(T$,2)<"24" AND RIGHT$(T$,2)<"60" THEN 140
 130 PRINT"Invalid - try again":GOTO 100
 140 h%=VAL(LEFT$(T$,2))
 150 m%=VAL(MID$(T$,4,2)):s%=0
 160 IF h%<>0 OR m%<>0 THEN 190
 170 INPUT "Stopwatch mode intended?"Y$
 180 IF Y$="N" OR Y$="n" THEN 100 ELSE IF Y$<>"Y" AND Y$<>"y" THEN PRIN
T"Y or N!":GOTO 170
 190 TIME=((h%*60+m%)*60+s%)*100
 200 REM * To run the current program
 210 *KEY0 RUN|O|M
 220 REM * To list with user choice of line numbers
 230 *KEY1 L.
 240 REM * To turn printer ON & scroll
 250 *KEY7 WIDTH72:|O|B|M
 260 REM * To turn printer OFF,no scrolling
 270 *KEY8 |C|N:WIDTH0|M
 280 REM * To display date and time
 290 REM * If you don't clear key first, may get BAD KEY on reloading
 300 *KEY9
 310 *KEY9 s%=TIME DIV100:m%=s%DIV60:h%=m%DIV60:CLS:P."* Date not known
at ";h%MOD24":";m%MOD60":";s%MOD60|M
 320 REM * adds date to key 9
 330 date=&B00+?&B09
 340 REPEAT
 350 date=date+1
 360 UNTIL ?date=ASC("*")
 370 date=date+1
 380 FOR I%=1 TO 14
 390 IF I%>LEN(D$)THEN ?(date+I%)=32 ELSE ?(date+I%)=ASC(MID$(D$,I%,1))
 400 NEXT I%
 410 REM * To add OLD to BREAK key
 420 *KEY10 OLD:|M
 430 END
```

Home Computer Show

Manchester

AT THE

Midland Hotel

PETER STREET

VISIT THE COMPLETE SHOW FOR THE HOME USER AND SEE:

A COMPLETE cross section of all hardware and software available to the home user.

A FULL RANGE of home computers priced from £50 upwards.

A COMPUTER ADVICE CENTRE run by independent experts for the answers to all your questions.

WIN WIN WIN TWO COMPUTERS — one for you, one for a school of your choice — to be won at each show: FREE entry form with advance tickets

April

THURSDAY 21 (12am–7pm)
FRIDAY 22 (10am–6pm)
SATURDAY 23 (10am–6pm)

ADULTS: £2.00 UNDER 8s & OAPs: FREE
25% DISCOUNT for parties of 20 or more.

COMING SOON

JUNE BIRMINGHAM

SPONSORED JOINTLY BY:

BBC Computing
Computing Today Personal Software
Personal Computing Today
Home Computing Weekly ZX Computing

Please enclose SAE
Home Computer Show
Send £1.00 Save £1.00 One person per coupon only
Mr. Mrs. Miss
Address

ASP Exhibitions
145 Charing Cross Road
London WC2H 0EE
01-437 1002

The Colourmaster

We bring you the ever-popular game Mastermind on your BBC Micro in a spectrum of colours.

Colourmaster will run on a Model A or Model B BBC Micro using the Mode 7 Teletext colour graphics. It is a computer equivalent of the well-known Mastermind (Mastermind is a tradename of Invicta Plastics) game. The user has the option of allowing the computer to generate a secret four colour sequence or specifying one. This allows two people to play each other or lets an individual try and beat the computer. The colours available are red, yellow, green, blue or white and are specified by the R, Y, G, B or W keys respectively. Each colour sequence input is terminated by Return.

After each input, the computer will display the specified colour sequence with related comments on its accuracy. The number of attempts to crack the colour code is limited to 10 and if you are unsuccessful, it will ultimately give you the correct sequence.

The program is structured and demonstrates the use of procedures and functions as can be seen in the program breakdown.

PROGRAM BREAKDOWN

Lines — **Description**

20–70 Array Declarations.

- CLR$ Holds the strings of available colours.
- OUT$ Holds the Teletext graphics characters in the available colours.
- COLOR% Holds the secret colour sequence as a numeric code (1-5).
- COLOR$ Holds the secret colour sequence in the related alpha characters.
- SEQUENCE% Holds the input colour sequence as a numeric code (1-5).
- SEQUENCE$ Holds the input colour sequence in the related alpha characters.
- C% Holds a working copy of COLOR%.

80–120 Set up the various characters relating to the available colours.

130–170 Set up the Teletext graphics strings in the available colours.

180–230 Display the operating instructions via the procedure 'PROCINSTRUCT' then clear the screen and request the user to state whether the secret colour sequence is to be specified or left to the computer.

240–260 Request the user to input the secret code and validates the input.

270–320 Copy the user specified colour sequence to the main arrays and set up the related character strings.

330–340 Generate a random colour sequence via the procedure 'PROCINIT' and clear the screen.

350–400 Form the main loop which is executed 10 times until an 'all correct' sequence is input.

410 Displays a failure message followed by the correct colour sequence.

420–440 Prompt the user for another game and acts accordingly.

450–540 PROCINSTRUCT — which displays the game instructions and awaits input of a space character to continue.

550–610 PROCINIT — which generates the random colour sequence and sets up the related alpha characters.

620–710 PROCINPUT — which requests the input of a colour sequence and validates the number of characters input.

720–790 FNVALID — this function validates the input colour sequence and returns an answer 'True' or 'False'.

800–890 PROCDISPLAY — which displays the input colour sequence specified.

900–1070 PROCCOMP — which compares the input colour sequence with the secret colour sequence and displays a summary of the result.

PROGRAM LISTING

```
 10 REM VERSION 2 16/1/83 J.SCOTT
 20 DIM CLR$(5)
 30 DIM OUT$(5)
 40 DIM COLOR%(4)
 50 DIM COLOR$(4)
 60 DIM SEQUENCE%(4)
 70 DIM C%(4)
 80 CLR$(1)="R"
 90 CLR$(2)="G"
100 CLR$(3)="Y"
110 CLR$(4)="B"
120 CLR$(5)="W"
130 OUT$(1)=CHR$(145)+CHR$(255)+CHR$(255)
140 OUT$(2)=CHR$(146)+CHR$(255)+CHR$(255)
150 OUT$(3)=CHR$(147)+CHR$(255)+CHR$(255)
160 OUT$(4)=CHR$(148)+CHR$(255)+CHR$(255)
170 OUT$(5)=CHR$(151)+CHR$(255)+CHR$(255)
180 PROCINSTRUCT
190 CLS
200 USERDEF=FALSE
210 PRINTTAB(0,23);"DO YOU WANT TO DEFINE THE SECRET CODE?";
220 ON INSTR("YN",GET$) GOTO 230,330 ELSE 220
230 USERDEF=TRUE
240 REPEAT PROCINPUT
250 UNTIL FNVALID=-1
260 USERDEF=FALSE
270 FOR I%=1 TO 4
280 COLOR%(I%)=SEQUENCE%(I%)
290 CHAR%=COLOR%(I%)
300 COLOR$(I%)=CLR$(CHAR%)
310 NEXT
320 GOTO 340
330 PROCINIT
340 CLS
350 FOR TRIES%=1 TO 10
360 REPEAT PROCINPUT
370 UNTIL FNVALID= -1
380 PROCDISPLAY
390 PROCCOMP
400 IF L%()4 THEN NEXT
410 IF L%()4 AND TRIES%=11 THEN PRINT:PRINT "  YOU HAVE FAILED MISERABLY !
":PRINT "  THE SEQUENCE WAS ":FOR I%=1 TO 4:PRINT COLOR$(I%);:NEXT:PRINT SPC(1
2);:TIME=0:REPEAT UNTIL TIME=400
420 CLS:PRINTTAB(0,23)"DO YOU WANT ANOTHER GAME ? ";
430 ON INSTR("YN",GET$)GOTO 190,440 ELSE 430
440 CLS:END
450 DEF PROCINSTRUCT
460 CLS
470 PRINTTAB(8,2) CHR$(141);"M A S T E R M I N D"
480 PRINTTAB(8,3) CHR$(141);"M A S T E R M I N D"
490 PRINTTAB(0,5)" This is the computer equivalent of the  well known MASTERMI
ND game."'" The game uses the following colours -"''
500 PRINT "    RED    specified by the R key"'"    YELLOW specified by the Y k
ey"'"    GREEN  specified by the G key"'"    BLUE   specified by the B key"'"
 WHITE  specified by the W key"''
510 PRINT " You have the option of getting the"'" computer to think of a FOUR
colour"'" sequence or specifying one yourself."'"Each colour sequence input is t
erminatedby RETURN."
```

J Scott

```
520 PRINT "You have a limit of 10 attempts"'"Press space bar to continue !"
530 REPEAT UNTIL GET$=" "
540 ENDPROC
550 DEF PROCINIT
560 FOR I%=1 TO 4
570 A%=INT(RND(1)*5+1)
580 COLOR%(I%)=A%
590 COLOR$(I%)=CLR$(A%)
600 NEXT
610 ENDPROC
620 DEF PROCINPUT
630 FLAG=FALSE
640 REPEAT
650 PRINTTAB(2,23);SPC(38);
660 PRINTTAB(2,24);SPC(37);
670 IF USERDEF=FALSE THEN INPUT TAB(0,23)"  INPUT NEXT COLOUR SEQUENCE ",SEQUE
NCE$ ELSE INPUT TAB(0,23)"  INPUT SECRET COLOUR SEQUENCE ",SEQUENCE$
680 IF LEN(SEQUENCE$)()4 THEN PRINTTAB(2,24);"INCORRECT NUMBER OF CHARACTERS I
NPUT";:TIME=0:REPEAT UNTIL TIME=200 ELSE FLAG=TRUE
690 UNTIL FLAG=TRUE
700 PRINTTAB(2,24);SPC(32);
710 ENDPROC
720 DEF FNVALID
730 FOR J%=1 TO 4
740 CHAR$=MID$(SEQUENCE$,J%,1)
750 FLAG=FALSE
760 CHAR%=INSTR("RGYBW",CHAR$)
770 IF CHAR%()0 THEN FLAG=TRUE:SEQUENCE%(J%)=CHAR%
780 IF FLAG=TRUE THEN NEXT ELSE PRINTTAB(2,24);"INCORRECT COLOUR SEQUENCE";:TI
ME=0:REPEAT UNTIL TIME=200
790 =FLAG
800 DEF PROCDISPLAY
810 S1%=SEQUENCE%(1)
820 S2%=SEQUENCE%(2)
830 S3%=SEQUENCE%(3)
840 S4%=SEQUENCE%(4)
850 PRINT TAB(0,TRIES%*2);OUT$(S1%);
860 PRINT TAB(5,TRIES%*2);OUT$(S2%);
870 PRINT TAB(10,TRIES%*2);OUT$(S3%);
880 PRINT TAB(15,TRIES%*2);OUT$(S4%);
890 ENDPROC
900 DEF PROCCOMP
910 L%=0:M%=0
920 FOR J%=1 TO 4
930 C%(J%)=COLOR%(J%)
940 NEXT
950 FOR J%=1 TO 4
960 IF SEQUENCE%(J%)=C%(J%) THEN L%=L%+1:C%(J%)=-1:SEQUENCE%(J%)=0
970 NEXT
980 FOR J%=1 TO 4
990 FOR K%=1 TO 4
1000 IF SEQUENCE%(J%)=C%(K%) THEN M%=M%+1:C%(K%)=-1:SEQUENCE%(J%)=0
1010 NEXT
1020 NEXT
1030 IF L%()0 THEN PRINTTAB(19,TRIES%*2);CHR$(135);L%;" OK AND IN POSITION"
1040 IF M%()0 THEN PRINTTAB(19,TRIES%*2+1);CHR$(135);M%;" OK OUT OF POSITION"
1050 IF L%=0 AND M%=0 THEN PRINTTAB(19,TRIES%*2);CHR$(135);"  ALL WRONG !"
1060 IF L%=4 THEN PRINT "CORRECT !! IN ";TRIES%;" ATTEMPTS":TIME=0:REPEAT UNTIL
TIME=300
1070 ENDPROC
```

Iris Generator Screen Dump

Learn how to draw patterns of all shapes and sizes and print them out too.

The program presented here generates patterns like a camera shutter, and it includes a screen dump routine for the Epson range of printers. The pattern generating part of the program operates as follows.

A polygon of the required number of sides is drawn on the screen. Then, a new polygon of the same number of sides is generated by choosing points the same distance from the corners of the existing polygon, and joining them together. This process is repeated until the polygon degenerates into a dot in the centre of the screen. The resulting effect is very pleasing, as can be seen from the example screen dumps.

The program as presented here runs in Mode 4, to ensure that Model A owners can run it. Model B owners can try to adapt the program to Mode 1 (for colour) or Mode 0 (for a higher resolution). In either case, the screen dump routine will have to be altered, as described towards the end of the article.

PROGRAM DESCRIPTION

Line Description

60-130
Allows the user to enter the number of sides of the polygon to be used in the pattern and the distance from each corner of the polygon at which the corners of the succedding polygon will be situated. S% should be greater than 3, so UNTIL statement at line 90 acts as a block on unreasonable values. No upper limit is imposed, but very large numbers of sides will result in a 'circle' being drawn not a polygon. MU should be between 0.5 and 0.99. A value of 0.5 makes each corner of the new polygon start between the old corners. For this reason, values of about 0.8 give the best results.

170 Turns the cursor.

180 Dimensions four arrays. The arrays X% and Y% will hold the co-ordinates of the corners of the polygon. L% and M% are used to generate the new co-ordinates in. After each generation, X% and Y% are updated from L% and M%.

190 Moves the cursor to the centre of the screen, because we are using a derivation of polar co-ordinates.

200-230
Generates the co-ordinates of the corners of the starting polygon.

240 Starts off the REPEAT loop which will only end when the polygon has degenerated into a dot in the centre of the screen.

250-280
Draws the polygon. The graphics cursor is moved to the last corner of the shape, then a loop is used to DRAW a line to the others.

290-340
Generates the new co-ordinates of the new polygon. The loop in line 290 goes through all the corners, but two corners are required for each new corner to be generated. Thus, J% is set to be the previous corner in line 300. If this gives an illegal corner (ie an element of the array with a subscript of zero) then the last corner is used instead (line 310). Lines 320 and 330 generate the X and Y co-ordinates the new corner.

350-380
Updates the arrays X% and Y% from L% and M%. If L% and M% were not used, X% and Y% would change half way through the previous loop, so destroying the algorithm's chances of working.

390 Checks to see if the polygon has got to the centre of the screen. If it has not, control passes back to line 250, otherwise the program falls through into the screen dump routine.

SCREEN DUMP ROUTINE

The Epson MX80F/T printer is capable of printing up to 480 dots across its width. These dots are sent in groups of eight, to be arranged down the page, as illustrated in Fig. 1.

However, the video RAM of the BBC Micro is arranged so that each byte is horizontal, rather than in the vertical format required by the Epson printer. To get around this problem, most screen dump programs use some cunning bit manipulation to arrange the bytes to be sent to the printer in the correct format. However, if the picture is printed *on its side*, the bytes of video RAM are automatically in the correct form for printing!

This arrangement also allows you to dump Mode 0 screens, which is impossible in the normal course of events, since the printer cannot support the 640 dots to a line required by such a program. Using the method above, only 256 dots are required across the page, since each mode has just 256 vertical points.

The precise video RAM arrangement of the BBC Micro is shown in Figs. 2 and 3. Each character position is simply divided into eight contiguous bytes (assuming Mode 4). Thus, the first byte of video RAM corresponds to the top byte (row) of the first character on the screen, the eighth byte of video RAM is the first byte of the

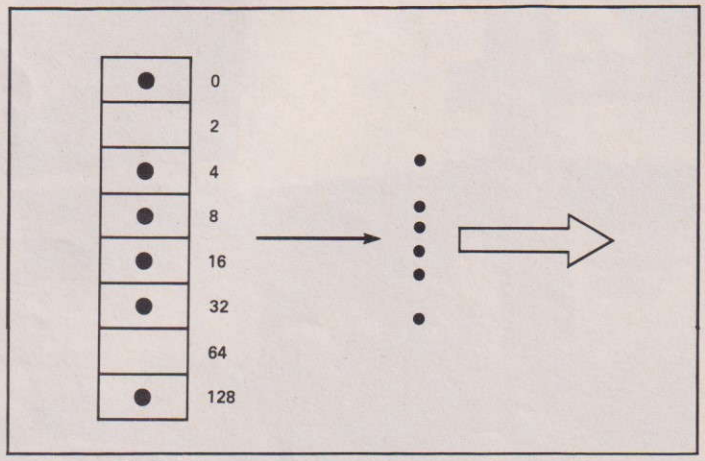
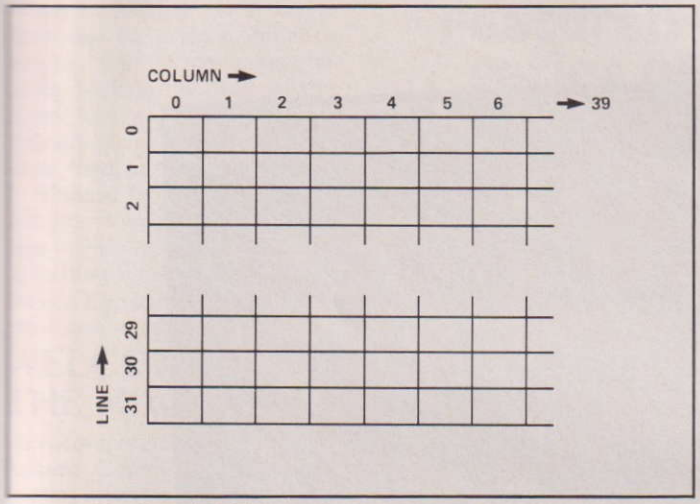
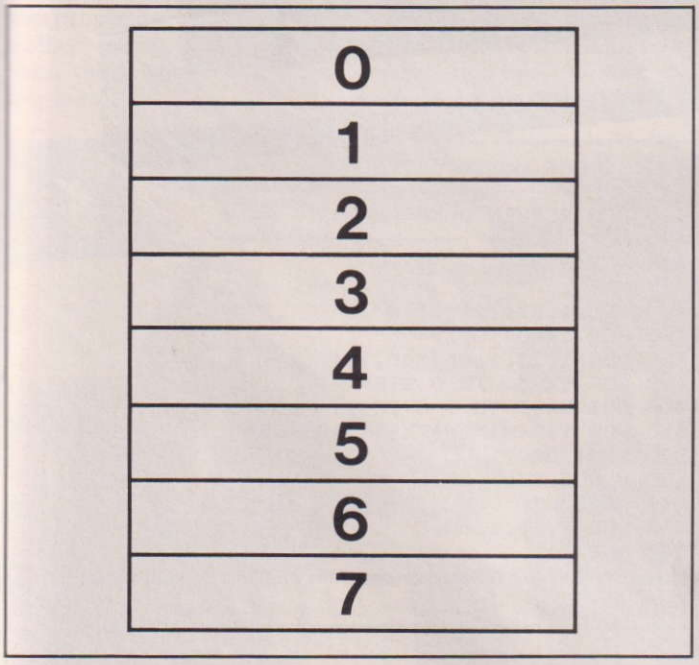


Fig 1. The MX80 F/T printer prints dots which are sent in groups of eight.

J Ruston



Figs 2 and 3. The precise video RAM arrangement of the BBC Micro.



second character on line one of the screen, and so on.

The Epson is also used with an Osborne 1, so it is not configured perfectly for the BBC Micro. Specifically, it requires Line Feed characters to be sent to it (code 10). The BBC Micro defaults to not sending out line feeds, so the *FX call number 6 is used to set the 'printer ignore character' to be null, to make the computer send the required Line Feeds. If your Epson is not adjusted in this way you can dispense with line 440. Similarly, line 530 sends a Line Feed to the printer. You may have to alter this line to VDU 1,13 to get a proper dump.

GRAPHIC DETAILS

Before the printer can receive high resolution data it must be initialized to do so, as in line 480. This instructs the printer to treat the next 256 characters it receives as graphics. The details are fully explained in the manual for the Epson.

Line Description

460 Removes the spacing the printer normally inserts between lines. Line 550 resets the spacing to normal. The '3' on the end of this line also de-selects the printer.

470 Starts a loop through all the columns of the display. This loop is the outer one because as the screen is going to be dumped on its side, the first line printed will consist of column 0, the next column 1, and so on.

480 Initialized the printer to receive 256 graphics bytes.

490 Loops through all the lines of the display. This is done backwards because the picture has the bottom edge of the left-hand side of the printer page.

500 Starts a loop through each of the eight bytes in each character position. This is also done backwards for the reasons outlined above.

510 Actually sends the required byte to the printer. The address of the byte is calculated using the index variables of the three loops so far introduced. T% is multiplied by 320 since each screen line takes up 320 bytes, and L% by eight because each character position accounts for eight bytes.

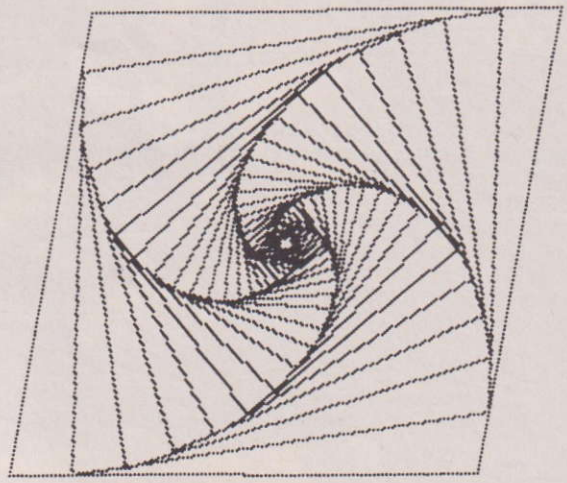
520 Closes up the byte loops required before a new line command can be issued.

530 Sends a sequence of bytes to the printer to get a new print line.

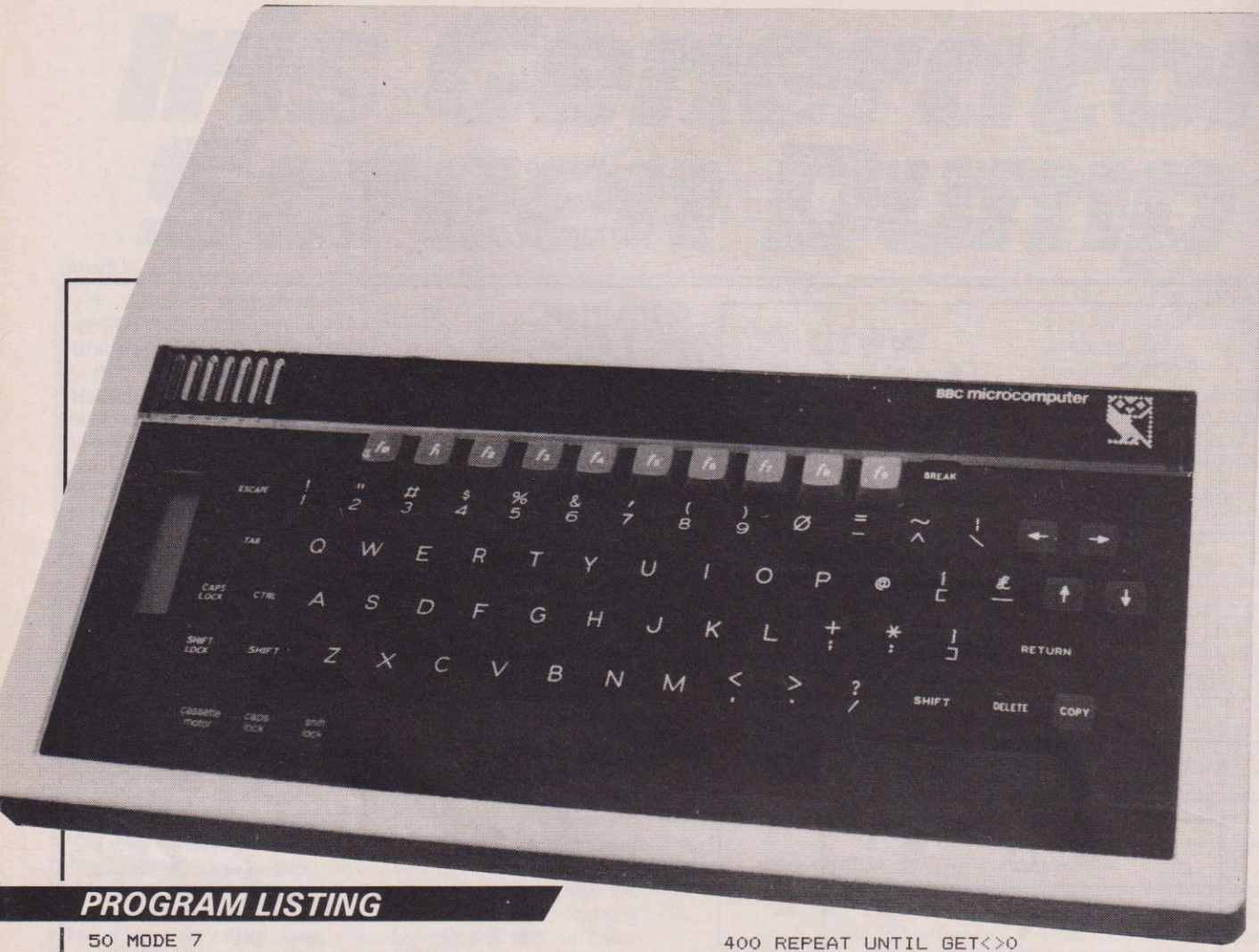
540 Ends the loop through all the character positions on the screen.

The other listing is a derivation of the routine for Mode 0.

Fig 4. A pattern generated by the Iris Generator program.



CONTINUED OVER



PROGRAM LISTING

```
 50 MODE 7
 60 REPEAT
 70 PRINT TAB(0,10);CHR$(131);STRING$(80," ")
 80 INPUT TAB(1,10)"Enter number of sides ",S%
 90 UNTIL S%>2
100 REPEAT
110 PRINT TAB(0,10);CHR$(129);STRING$(80," ")
120 INPUT TAB(1,10)"Enter 'MU' ",MU
130 UNTIL MU>0.5 AND MU<1
140 RMU=1-MU
150 MODE 4
160 PRINT S%;" sides, MU is ";MU
170 VDU 23;8202;0;0;0;
180 DIM X%(S%),Y%(S%),L%(S%),M%(S%)
190 VDU 29,640;512;
200 FOR T%=1 TO S%
210 X%(T%)=SIN(RAD((360/S%)*T%))*450
220 Y%(T%)=COS(RAD((360/S%)*T%))*450
230 NEXT T%
240 REPEAT
250 MOVE X%(S%),Y%(S%)
260 FOR T%=1 TO S%
270 DRAW X%(T%),Y%(T%)
280 NEXT T%
290 FOR T%=1 TO S%
300 J%=T%-1
310 IF J%=0 THEN J%=S%
320 L%(T%)=MU*X%(T%)+RMU*X%(J%)
330 M%(T%)=MU*Y%(T%)+RMU*Y%(J%)
340 NEXT T%
350 FOR T%=1 TO S%
360 X%(T%)=L%(T%)
370 Y%(T%)=M%(T%)
380 NEXT T%
390 UNTIL ABS(X%(1))<10 AND ABS(Y%(1))<10
400 REPEAT UNTIL GET<>0
410 REM ------------------------
420 REM SCREEN DUMP
430 REM ------------------------
440 *FX 6,0
450 VDU 2
460 VDU 1,27,1,65,1,8
470 FOR L%=0 TO 39
480 VDU 1,27,1,75,1,0,1,1
490 FOR T%=31 TO 0 STEP -1
500 FOR G%=7 TO 0 STEP -1
510 VDU 1,?(HIMEM+T%*320+G%+L%*8)
520 NEXT G%,T%
530 VDU 1,13,1,10
540 NEXT L%
550 VDU 1,27,1,50,3
560 END
```

Listing 1. The Iris Generator program and the screen dump routine.

```
460 *FX 6,0
470 VDU 2
480 VDU 1,27,1,65,1,8
490 FOR L%=0 TO 79
500 VDU 1,27,1,75,1,0,1,1
510 FOR T%=31 TO 0 STEP -1
520 FOR G%=7 TO 0 STEP -1
530 VDU 1,?(HIMEM+T%*640+G%+L%*8)
540 NEXT G%,T%
550 VDU 1,13,1,10
560 NEXT L%
570 VDU 1,27,1,50,3
580 END
```

Listings 2. The screen dump routine in Mode 0.

The BBC Programme - a spotlight on Richard Gomm

Featured in the first series of the BBC's Microcomputer programmes, Richard Gomm's story is one of computing's success stories.

When the BBC producer David Allen was gathering material to feature in the micro computer series 'Making the Most of the Micro' he was anxious to have ordinary people using micros for more than just playing games.

Richard Gomm, seen in the first programme of the series, was ideal in this respect. Not only was he getting the real benefit from using a micro, but he also had a great deal to say about it.

WELCOME TO THE MACHINE

Microcomputers, according to Richard Gomm, are to further existing interests: "The danger is that in some cases the micro becomes an end, not the means. I want people to see a micro as just another piece of equipment which they might use in their daily lives".

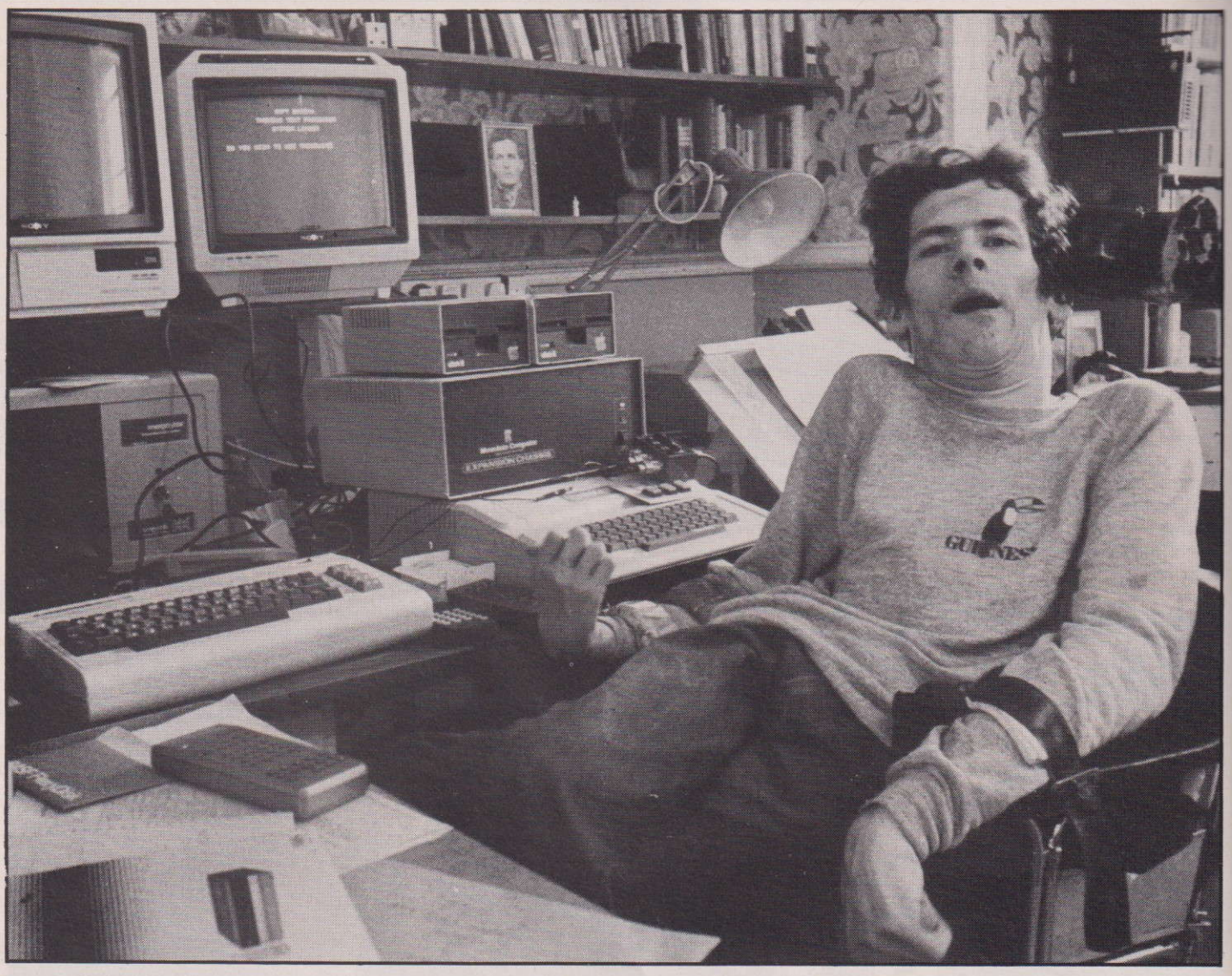
Thirty year old Richard Gomm became aware of the potential of computers while studying philosophy at York University. Severely disabled due to an accident at birth, he was writing essays using a rod attached to a band around his head—the only part of his body he can control—to press the keys of an electric typewriter. This was a slow and laborious process that made no allowance for mistakes or second thoughts. Friends suggested a computer could help but the best option was rather large and unfriendly, so the idea was not very practical.

However, in 1979 the situation changed. Computers became smaller and cheaper. A new era of computing was beginning. Now living with his parents in West Wales, Gomm was faced with the task of producing a thesis of around 60,000 words of closely argued philosophical analysis for a Ph.D at Swansea University. Although he had no knowledge of computing, he got himself a micro on which he could run a word processing package. As he put it, "The freedom this gave me was marvellous! Now I could get my work in the precise form I wanted before printing it letter perfect. This did not mean my work was necessarily any better—I just had no excuses now!"

BREAKING THROUGH

Although a breakthrough, the software did not do all he would have liked and, knowing it

CONTINUED OVER



was unlikely that any other package would meet his rather specialized needs, he set about writing his own. Learning everything from books and manuals, his 'Theseus Text Editor', as he calls it, took a year to write. His father, Mervyn Gomm, remembers how it was at the start, "It was terribly frustrating for him. The first few weeks we'd go into his room and Richard would be bathed in sweat and up on the screen would be 'Syntax Error'. We'd go in at nine o'clock at night to see if he was ready for bed and he'd say 'No, I've got to put a bit more in'. He really put his mind to mastering that computer."

After twelve months of effort he had a word processor with the facilities he needed to make writing easier. Its major advantage was the definable vocabulary of up to 780 words which could be placed in the text with the minimum of key-strokes. For example, by pressing '4' and then 'B' he would get the word 'before'. The 50K program would even print out his name in a form that imitated ordinary handwriting, giving him a 'signature' for personal correspondence.

His mother, Kathleen Gomm noticed the difference it has made to his writing, "I'm not sure that he actually does more writing but he's more satisfied with the writing he does because by the time he prints it out, it's exactly what he wanted to say. Whereas in the old days mistakes would remain because of the sheer physical difficulty of typing".

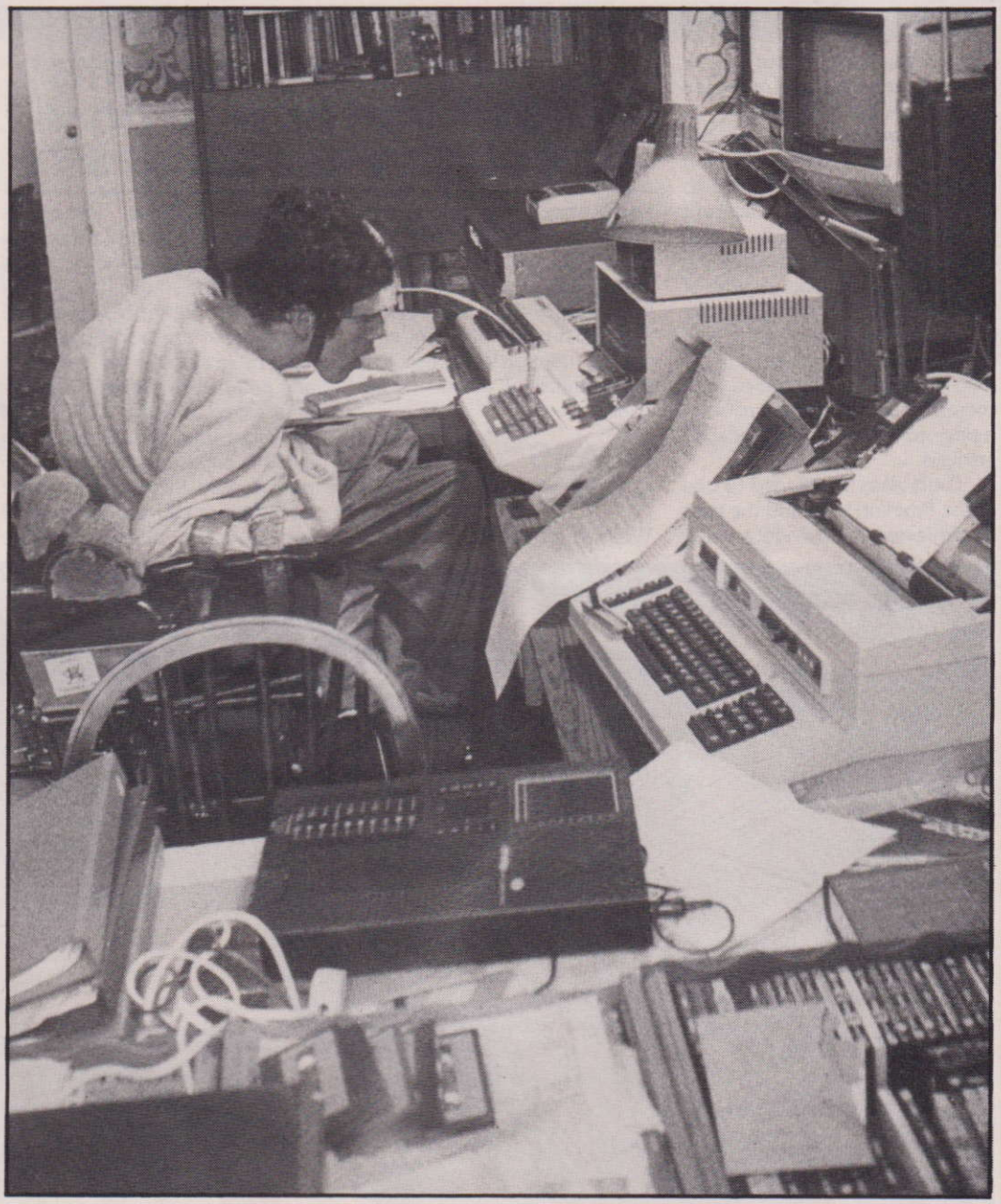
IN CONTROL

Word processing was just the first of the jobs he had in mind for his computer. The next was to try to achieve some form of control over his electrical equipment. He had seen an advertisement for 'mains relays' that could be switched by computer. "My idea was to gain control over my lamp, television, radio and so on via my micro. Suddenly I did not have to ask

people to switch the lamp or printer on. I no longer had to persuade people to watch the late movie with me; my software would turn off the TV after it had finished. I could control my environment!"

The software he wrote would control up to six mains devices, switching them on or off at any preset time. Commands are entered either through the keyboard or by pressing a pressure pad with a corresponding visual display on a second monitor. As the system would be in use twenty four hours a day, seven days a week, he uses a second smaller micro to run the software, thus leaving his main 48K RAM computer free for further software development, word processing or games.

Although he is reasonably satisfied with his programming achievements and acknowledges that they have extended his capabilities as a writer as well as giving him an extra degree of independence, he knows that micros are only part of the solution to the problems of disabled people. After watching a television documentary film where someone similarly disabled was provided with a voice through a microcomputer and was then seen using it to order a pizza over the telephone he commented "It's one thing to be able to order a pizza but it's another to be able to open the door to go get it."



THE PEN IS MIGHTIER . . .

Gomm hopes one day to provide his own solutions to problems like that by combining his skills as a software writer with his intimate knowledge of the needs of the disabled. An example of this is his own difficulty in using the conventional computer keyboard which places a great deal of strain on his back. To overcome this he envisages a computer system operated not by a keyboard but by some kind of input device that could work over a distance, perhaps some kind of light pen: "My idea would then be to have two screens. One screen would have menus and alphabets etc.—in a way this screen would be the keyboard. The other screen would act as a normal screen. Thus you would point to letters and words on one screen and they would be registered on screen two as if you had typed them in."

Micros may turn out to be the means to achieve what he wants, but for Richard Gomm they will never be an end in themselves. He has already published two volumes of his poems and hopes to do more in the near future. And there is also his weighty thesis on his hero, the philosopher Ludwig Wittgenstein

It was Wittgenstein who said "The limits of my language are the limits of my world". Half a century later Richard Gomm could say the same about machine code and BASIC.

Experimental Data Collection

Collecting data for a survey? This programs shows you how to do it thoroughly yet painlessly.

This program might be useful for anyone using a BBC in connection with a survey or experiment where the frequency of events is being studied.

The program sets up the computer as a data collection device which can record up to 2,000 events in 10 different classes and then write a file where each record contains the class of an event and its time of occurrence. Obviously, the subsequent treatment of the data is very much dependent on the kind of event being studied and, therefore, no attempt has been made to provide methods of analysis.

Each class is assigned to one of the function keys and the user is requested to provide a short description. This description is displayed on the top half of the screen to act as a reminder during the data collection. The lower half of the screen displays the total number of events so far, the last function key pressed and the number of events in each class.

CLASS STRUGGLES

When the user has entered the class descriptions, Return is pressed and data recording starts; whenever a function key is pressed, the time (taking the pressing of Return as zero) and key number are stored in arrays. The program can be halted at any point by entering 's' and the data saved on tape. As it stands the program can deal with up to 2,000 events, though there is no reason why this should not be increased or decreased provided line 110 is also changed; this line acts as a safety valve to prevent an over zealous experimenter trying to record more events than the arrays can store!

As an example, a traffic survey could be undertaken (taking your power supply from the nearest lamp post!) where the class descriptions could be car, bus, van, lorry, motor bike and bicycle. This would use function keys 0 to 5 only and the data file could be used to provide information on traffic density and type over a period of time.

Naturally, it is important to read back the data file in the same format as it was written and the lines 580 to 630 should be mirrored in the file reading procedure—with appropriate changes, of course.

R Grubb

PROGRAM LISTING

```
 10 DIMCLOCK(2000),FKEY%(2000),DESC$(9),SUM%(9)
 20 PROCsetup
 30 REPEAT
 40   A=INKEY(0)
 50   IF A=-1 GOTO40
 60   IF A=ASC("s") PROCsave:DONE=TRUE
 70   IF A>96 AND A<107 PROCstore
 80   UNTIL DONE
 85 END
 90  :
100 DEFPROCstore
110 IF N=2000 PROCoverflow:GOTO 210
120 N=N+1
130 CLOCK(N)=TIME
140 SUM%(A-97)=SUM%(A-97)+1
150 FKEY%(N)=A-97
160 PRINTTAB(0,18)CHR$133"Totals so far :"
170 PRINTTAB(0,16)CHR$130; N+1" EVENTS.         LAST EVENT = ";FKEY%(N)
180 FORJ=0 TO 4
190   PRINTTAB(0,19+J)CHR$130"KEY ";J " = ";SUM%(J),TAB(20,19+J)"KEY ";J+5," =";SUM%(J+5)
200   NEXT
210 ENDPROC
220 :
230 DEFPROCsetup
240 CLS
250  PRINT''"Enter a brief description of the"'"event assigned to each function key"''
260 FORI=0 TO 9
270   SUM%(I)=0
280   PRINT"KEY ";I" ";:INPUTDESC$(I)
290   DESC$(I)=LEFT$(DESC$(I),25)
300   NEXT
310 N=-1:DONE=FALSE
320 INPUT''"PRESS RETURN TO START RECORDING"A$
330 CLS
340 PRINT''
350 FORI=0 TO 9
360   PRINTCHR$131"KEY ";I;"_"DESC$(I)
370   NEXT
380 PRINT'CHR$133"   's' will terminate recording"
390 PRINTCHR$134STRING$(38,"*")
400 *KEY0 a
410 *KEY1 b
420 *KEY2 c
430 *KEY3 d
440 *KEY4 e
450 *KEY5 f
460 *KEY6 g
470 *KEY7 h
480 *KEY8 i
490 *KEY9 j
500 TIME=0
510 ENDPROC
520 :
530 DEFPROCsave
540 CLS
550 INPUT''"Enter file name "A$
560 A$=LEFT$(A$,10)
570 IF A$="" GOTO540
580 X=OPENOUT(A$)
590 PRINT£X,N
600 FORI=0 TO N
610   PRINT£X,CLOCK(I),FKEY%(I)
620   NEXT
630 CLOSE£X
640 ENDPROC
650 :
660 DEFPROCoverflow
670 FORI=0 TO 13:PRINTTAB(0,I)SPC(39):NEXT
680 PRINTTAB(5,7)"NO MORE EVENTS CAN BE RECORDED."TAB(5,8)"PLEASE SAVE YOUR DATA NOW."
690 ENDPROC
```

BOOKSH

Buying a book to suit your needs can be risky. We help you decide which one.

When a new machine becomes available on the market, there inevitably follows a deluge of books telling us how to use the machine to its best advantage. The BBC Microcomputer is certainly no exception and we take a look here at some of those books.

No doubt more books will keep appearing on the publishers' lists as more people investigate and experiment with the BBC Micro so don't assume that this list ends here!

The BBC Micro Revealed: If you've mastered the contents of the manual that came with your BBC Microcomputer and now want to continue your exploration of the computer's functions and capabilities, this book is for you. The author, a 17 year old student, spent months delving into the computer's internal operations in order to reveal a large number of sophisticated techniques to help the reader improve his or her programming skills.

The book includes the following features: details of how to construct your own display modes; a way to scroll the display in any direction (up, down, sideways and even diagonally); a visual analogue of the computer's memory transactions; information on the way in which the computer stores its programs and line numbers; a technique for increasing the speed of your programs by up to 10%; instructions on how to pass arrays and matrices to user-defined functions and procedures; and much, much more. If you're serious about developing your programming skills on the BBC Micro, this book will prove an invaluable aid.

The BBC Micro Revealed by Jeremy Ruston is published by INTERFACE at £7.95 for 144 pages.
ISBN 0 907563 15 5.

Learning to use the BBC Microcomputer: This is one of a new series of 'Learning to use' books, designed to provide potential users, established users, teachers, students and businessfolk with standardised introductions to use of popular microcomputers. This beginner's guide really does begin at the beginning: it assumes that you want to learn how to use the BBC Microcomputer in your work or leisure, rather than become a theorist in computing. The book provides a simple, down-to-earth, jargon-free introduction to the machine and its software.

Many applications of the BBC Microcomputer are described including business, educational and hobby uses; the micro's ability to produce and draw pictures and diagrams is explored and explained; and programs for a large number of graphics applications are presented. The book will not only appeal to the new BBC Microcomputer owner but also to potential buyers since it will tell them how the BBC Microcomputer operates and performs and will help them assess whether the machine will suit their needs.

Learning to use the BBC Microcomputer by P N Dane is published by Gower Publishing Company Limited at £4.95 for 84 pages.
ISBN 0 566 03452 2.

The Computer Book: This publication, although produced by the BBC, has no direct connection with either the BBC Micro or **The Computer Programme.** It simply attempts to introduce the 'computer shy' individual to the wide and diverse subject of computing, covering the ground in a relaxed friendly fashion. The book doesn't introduce any radical new ideas and certainly cannot be regarded as a 'text' on computers but then that isn't really its object.

The features which really make this book stand out are its excellent production and layout and the clever use of photographs, illustrations and cartoons to keep the reader both interested and amused. If the quality of the editorial content was to the same standard . . .

The Computer Book by Robin Bradbeer, Peter de Bono and Peter Laurie is published by BBC Publications at £6.75 for 254 pages.
ISBN 0 563 16484 0.

HELF

30 Hour BASIC: If you want to approach the subject of computer programming in a disciplined and methodical fashion then this book, which has been produced in conjunction with the BBC series, is almost certainly a recommended buy. You don't really need a micro to complete the course, although one would be helpful, and the book is not specifically related to the BBC Micro . . . a special version is also available for the Sinclair ZX81.

If you like your approach to computers to be light-hearted, this book will probably not appeal. Also, its approach means that as well as learning the ins and outs of BASIC programming, you will learn to write clear and logical programs, something that happens all too seldom.

30 Hour BASIC by Clive Prigmore is published by The National Extension College at £5.50 for 256 pages.
ISBN 0 86082 269 9.

Practical Programs for the BBC Computer and the Acorn ATOM: This somewhat slimmer volume contains four chapters based around a number of simple programs which are reproduced for both the BBC Micro and the ATOM. The presentation and layout is excellent and the structure of each of the examples is clearly explained. However, the real meat is to be found in the fifth chapter which presents SPL, Simple Programming Language, a new compiler for both types of micro. As well as providing a second high-level language, this chapter demonstrates how to go about writing a simple compiler.

Practical Programs for the BBC Computer and the Acorn ATOM by David Johnson-Davies is published by Sigma Technical Press (distributed by John Wiley & Sons Ltd) at £5.95 for 120 pages.
ISBN 0 905104 14 5.

Assembly Language Programming for the BBC Microcomputer: Every BBC Micro comes equipped with an immensely powerful and very fast asembler; assembly language statements and BASIC statements can be freely mixed which enhances the programmer's potential control over the machine.

This book assumes that the reader is proficient in BASIC, but assumes no knowledge of assembler at all, taking you step by step from the basics to complex implementation.

The book is aimed at three types of reader: current owners of BBC Micros who want to extend their knowledge into machine code; the teacher or student of Computer Science who wants to use this text as part of a structured course; and those already experienced BASIC programmers who are wondering whether to buy the BBC Microcomputer.

Containing 73 listings of programs, the book is completely self-contained, and has various appendices on the 6502 instruction set, floating point and the user port, and a section on combining programs in the BBC computer using PAGE and *LOAD.

Two companion tapes are also available with the book if you feel you do not want to type in all the programs yourself.

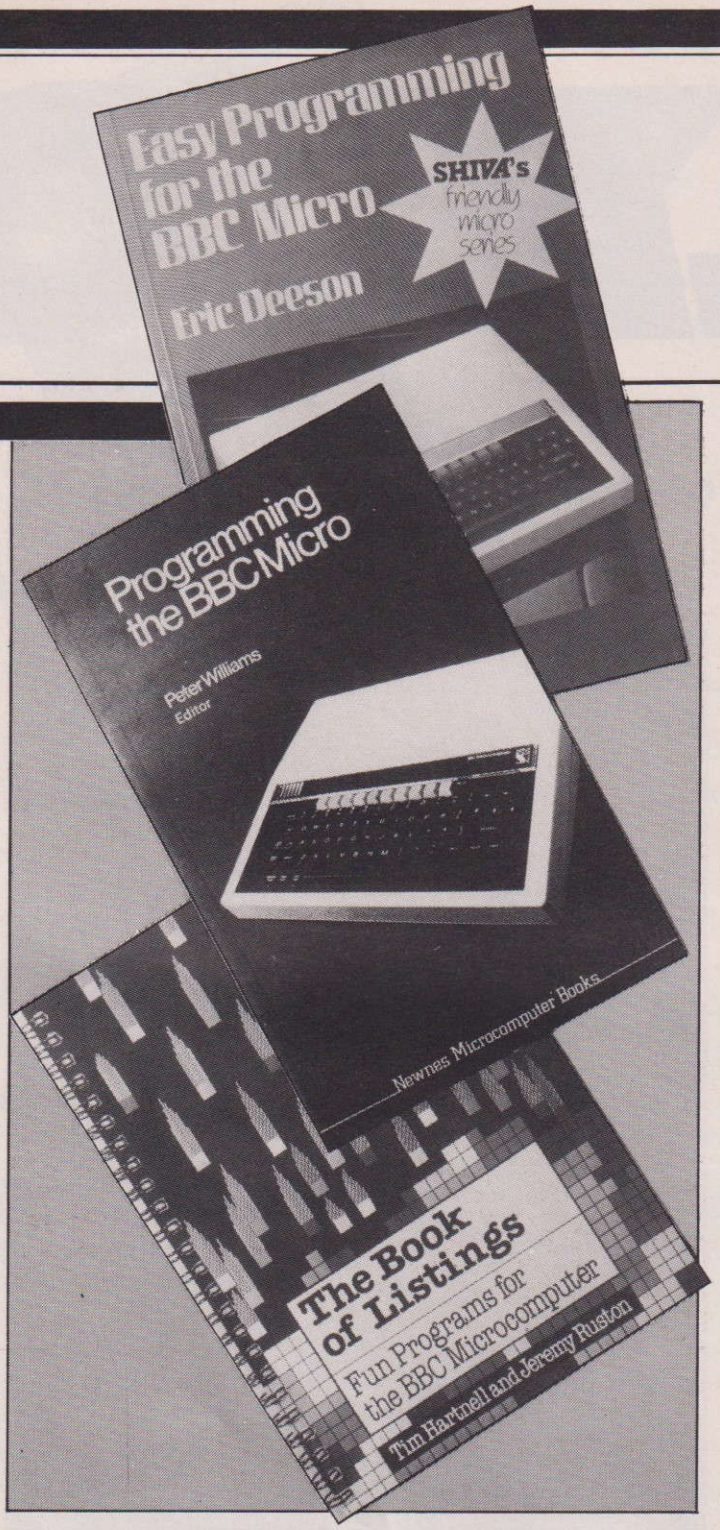
Assembly Language Programming for the BBC Microcomputer by Ian Birnbaum is published by Macmillan Press at £8.95 for 305 pages. The cassettes are £9.00 each or £16.00 for two.
ISBN 0 333 34585 1

BASIC Programming on the BBC Microcomputer: To have produced this introductory book in so short a time is a remarkable achievement by both the authors and the publishers. The sad fact, however, is that this is an introductory book, and as such tends to leave you waiting for more. The volume was put together with the assistance of Acorn, the company who designed and produced the BBC Micro, so it is very specifically related to that product.

The book is practical in nature with lots of small examples to try out and problems to solve. The actual information content is not significantly more than that in the early version of the **User Guide** except that the facts have been arranged in a more readable form. Although the book has areas of weakness, it does stand up as an introduction but one hopes that the second volume will not be too long in coming.

BASIC Programming on the BBC Microcomputer by Neil and Pat Cryer is published by Prentice Hall International at £5.95 for 205 pages.
ISBN 13 066407 3.

CONTINUED OVER



Programming the BBC Micro: This book, written for the user of the BBC Micro, concentrates on the programming and application of the machine, assuming little or no knowledge on the reader's part. Early chapters introduce some practical points on programming the micro including functions, subroutines, procedures, program development and program libraries, etc.

Following chapters cover features of BBC BASIC as applied to graphics, words, numbers, and the machine's sound facility. The book covers slightly more advanced topics such as machine code, Hex, assembly language programming, interfacing and file handling. The final chapter looks at hardware, and appendices list the hardware specification, error codes, ASCII codes and the 6502 instruction set.

Programming the BBC Micro by John D Ferguson, John Gordon, Louie Macari, Simon Rushbrook Williams and Anthony Shaw, and edited by Peter Williams is published by Butterworth & Co at £6.50 for 167 pages.
ISBN 0 408 01302 8.

Let Your BBC Micro Teach You to Program: This book is the ideal companion for you if the BBC Micro is your first computer. It takes you step by step through programming in BBC BASIC (giving a number of worthwhile programs) and then advances to help you develop and expand your programming skills gradually.

Once you have mastered the fundamentals of BASIC, this book will help you understand and apply such things as the use of the ENVELOPE command, how to master the graphics and use them to best effect, the use of VDU drivers, graphics windows, how to define your own character, the use of functions and procedures, and ways of writing better programs. There are a number of utility and games programs in the book, ready to run.

Let Your BBC Micro Teach You to Program by Tim Hartnell is published by INTERFACE at £6.45 for 193 pages.
ISBN 0 907563 14 7.

The Book of listings. Fun Programs for the BBC Microcomputer: This first BBC book of listings contains a host of games and other programs, ranging from arcade-like action programs, through board games which will tax your wits, to some startling graphics demonstrations.

The authors have tried to make the most of the colour and sound potential of the BBC Micro, writing programs to run on both models.

The programs were developed on both the Model A and B machines with the 0.1 Operating System.

Structured programming techniques have been used as far as possible. Although programs may thus be a little longer than strictly necessary, they do tend to be relatively easy to debug and modify. Many of the program notes include suggestions as to how you can adapt the programs to make them your own and to develop them further.

The programs are intended to entertain and to teach useful programming techniques.

The Book of Listings. Fun Programs for the BBC Microcomputer by Tim Hartnell and Jeremy Ruston is published the the British Broadcasting Corporation at £3.75 for 156 pages.
ISBN 0 563 16534 0.

Easy Programming for the BBC Micro: This book is explicitly a beginner's guide to working with the BBC Microcomputer. Starting with an explanation of what a computer is, the author takes you through the complexities of BBC computing including animation, strings, the use of flowcharts, editing, arrays, the comprehensive sound capabilities of the BBC Micro and includes a case history of a bugged program. Included in the text are 28 complete and ready to run programs and another 12 'additional programs' are listed at the end of the book which can be copied and RUN at any stage.

The book was written before the full BBC Manual was available but it is suggested that the two should be used in conjunction.

Easy Programming for the BBC Micro by Eric Deeson is published by Shiva Publishing Limited at £5.95 for 128 pages.
ISBN 0 906812 21 6.

Alternative Printout

Tandy's Colour Graphics Printer provides you with a useful, cheap alternative for the BBC Micro.

Even though the Color Graphics Printer (CGP-115) is really a Tandy computer peripheral, it has a standard Centronics-type parallel interface and surprisingly, for such a low cost printer, a RS232 serial interface as well. Therefore, the printer should work with most of the personal computers on the market that have a printer port. The means the BBC Micro.

CHEAP AT THE PRICE

For those people looking for a low cost printer for program listings and the like rather than for high quality word processing printout, then the CGP-115 must be a printer worth considering. The cost of this printer (£149 inclusive) makes it cheaper than say the Seikosha GP range, the device also boasts the obvious added attraction of the excellent graphic capability (as well as text) and the choice of four colours—red, blue, green and black. The CGP-115 is an X-Y plotter rather than a conventional printer and has many features which make it unique. it prints out on four and half inch roll of paper but does have the ability to print 40 or 80 characters per line in text mode; with 80 characters per line a magnifying glass is optional! With printer commands like move, draw, rotate text and selectable character size it has much potential and will be found in many different applications as well as for the computer hobbyist.

NOW FOR THE GOOD NEWS

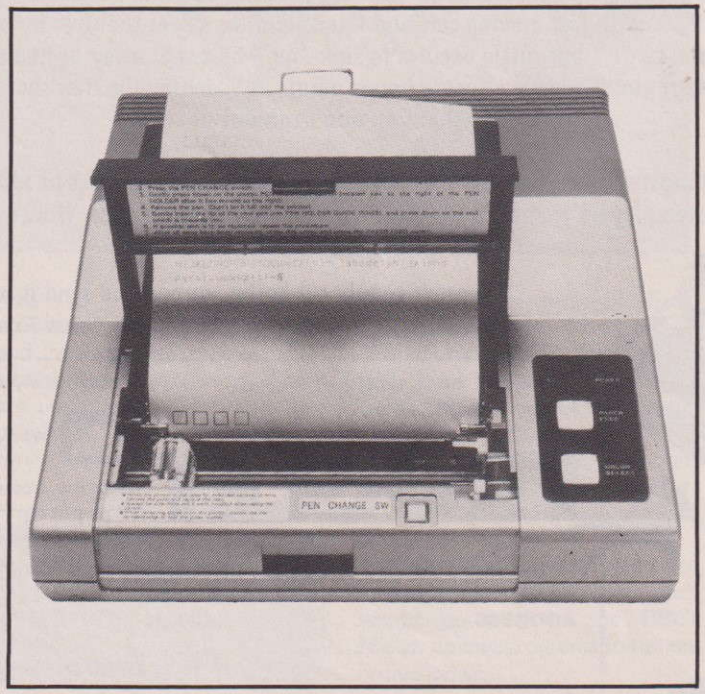
The CGP-115 has been well thought out and designed (needless to say it has a 'Made in Japan' sticker on the case); for example, it has not one or two but three different and selectable power-on function tests!

The good news for the BBC Micro owners is that no hardware modifications are required to connect the BBC machine to the printer via the parallel printer port. The software which controls the printer, however, has to be in a slightly different format to that of the information and software examples in the manual. The manual assumes, not suprisingly, a TRS-80 or a Tandy Color Computer will be used.

It is not possible when using the CGP-115 with the BBC Micro to use the PRINT CHR$ (number) statement on many of the commands, as the screen VDU driver will interpret (number) as a VDU graphic or colour definition command. However, there are several other ways of sending instructions to the printer from a program. Using the VDU 1, (command) statement (which in many respects if the same as CHR$) will send the next instruction only to the printer. Should you not be familiar with the VDU commands please refer to the relevant chapter in the BBC Micro's **User Guide**.

AT YOUR COMMAND

Some of the printer commands require to be terminated with a Carriage Return. It is advisable to insert VDU 1, 13 (Carriage Return = 13) after each printer command even if it may not always be necessary. The ASC and PRINT commands can also be used with VDU 1 for legibility. For example, (VDU 1,65), (VDU 1, ASC"A") and (VDU 1, : PRINT "A"), all have the same effect of sending "A" (return) to the printer (ASCII A=65). This will reset the pen position and put the printer into text mode. It is best to adopt a particular convention about the format of these commands to avoid future difficulties.



This program generates patterns on the CGP-115, similar to the pattern shown on the box which the printer is packaged in. The program also displays the same pattern on the TV while the printer is plotting out; the colours, will be different.

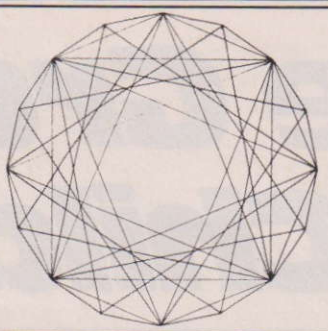
The program prompts for the number of points and assigns this value to variable N%. The program takes N% equidistant points of a circle (lines 10 - 130) and puts the values into arrays X% and Y%. The program will draw a straight line from one point on the circle right around the points, missing out either one, or two, or three points until it reaches the point it started from. The next colour is used and the process is then repeated until all the points on the circle have been connected.

Lines 140 to 150 initialise the printer for graphics mode and to plot using 'solid' lines (there are 15 different line types from solid to dashes to dots!). Line 160 makes sure there are an even number of points on the circle. C% is used for selecting the printer pen colour. The procedure definitions (lines 240 and 250) send the command 'M' for move to the printer with the three necessary values, and likewise the 'D' for draw command. Lines 160 to 290 draw the pattern onto both the screen and printer.

The printer moves are draw command are very similar to the BBC Micro's VDU command. By manipulating the equations in lines 100 and 110, different types of patterns can be created.

SPEEDING UP

When I first used the program I

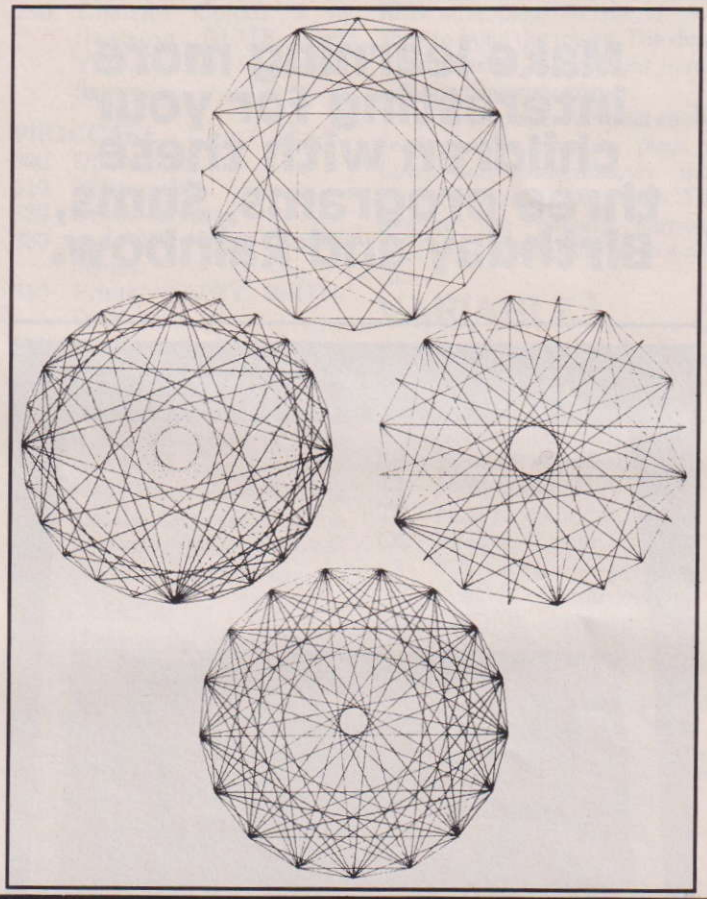


R Thomas

was delighted with the speed increase, one screen now taking less than two minutes, just enough time to go and put the kettle on. However, I was loading the screen dump program after running the graphics routine and this meant I had all kinds of messages appearing on the screen, such as Loading and RUN.

The final problem, then was to merge in some way the two programs, so that typing RUN would start the graphics program and then continue with the printer program. It's quite easy to do this when you recall how the BBC Micro stores its programs internally. A BASIC program is stored from a location given the name PAGE by the operating system, and to start with PAGE is location 3584 in decimal. Once the program is in the memory, the first free location above it is given the name TOP. You might think, therefore, if you now reset PAGE with:

PAGE=TOP



The second program would start loading from the first free location above the first one. Unfortunately, it's not quite so simple. For reasons best known to itself, the variable PAGE can only take values which are multiples of 256, so a few more steps are needed; here's a simple way of doing it:

. . . load the graphics program.
. . . find how long it is by typing PRINT TOP (suppose TOP is 4171).
. . . find the next highest number divisible by 256 (this is 4352).
. . . add two more lines to the graphics program.
nnn PAGE = 4352 (or whatever)
mmm RUN
. . . make sure your screen dump has higher line numbers than mmm.
. . . type PAGE = 4352.
. . . load the screen dump program.
. . . type PAGE = 3584 and RUN.

It's as simple as that, but don't forget to switch the printer on!

PROGRAM LISTING

```
10 REM WRITTEN BY R.& A. THOMAS
20 REM FOR THE CGP-115 PRINTER
30 DIM X%(100),Y%(100)
40 CLS
50 INPUT" HOW MANY POINTS ?"N%
60 FOR I%=1 TO N%
70 REM PUT X & Y CO-ORDINATES
80 REM FOR "N%" POINTS OF A
90 REM CIRCLE INTO AN ARRAY
100 X%(I%)=SIN(2*PI*I%/N%)*512+640
110 Y%(I%)=COS(2*PI*I%/N%)*512+512
120 NEXT
130 MODE 2
140 VDU 1,18:REM PRINTER GRAPHICS MODE
150 VDU 1,ASC"L",1,0
160 FOR I%=1 TO N%/2-.5
170 GCOL 0,I% MOD 4:REM VDU COLOUR
180 VDU 1,67,1,(I%-1) MOD 4 +48,1,13
190 C%=1: REPEAT
200 MOVE X%(C%),Y%(C%)
210 D%=C%+I%
220 IF D%>N% THEN D%=D%-N%
230 DRAW X%(D%),Y%(D%)
240 REM MOVE AND DRAW FOR PRINTER
250 PROCPCHAR("M",X%(C%)/3,-Y%(C%)/3)
260 PROCPCHAR("D",X%(D%)/3,-Y%(D%)/3)
270 C%=C%+I%
280 IF C%>N% THEN C%=C%-N%
290 UNTIL C%=1
300 NEXT
310 VDU 1,ASC"A",1,13
320 END
330 DEF PROCPCHAR(A$,N%,M%)
340 A$=A$+STR$(N%)+","+STR$(M%)
350 FOR N%=1 TO LEN(A$)
360 VDU 1,ASC(MID$(A$,N%,1))
370 NEXT
380 VDU 1,13
390 ENDPROC
```

Simple Graphics for Children

N Stanger

Make learning more interesting for your children with these three programs, Sums, Birthday and Rainbow.

SUMS

This is to show how graphics with user defined characters can improve a simple program.

The title is displayed, and a sequence of a goat crossing a river. The child is invited to help the goat cross the river by getting all the sums right.

KIDS STUFF

The child is asked to add two to a number between one and 10. If the answer is correct, there is a sound signal and the goat proceeds; the goat stays put.

If the 10 questions are answered correctly the goat eats some grass on the other side of the river, and the program re-runs itself with an increased base number. If any answers are wrong, the child is given more practice at the same level.

VARIABLES

T% Question number.
N% Base number.
F Number to add.
I% Loop variable.
X% Horizontal position of picture.
Y% Vertical position of picture above 20.
S% User defined picture.
Q Delay factor.
11,12 Vertical limits of screen to be cleared.
Q%(F) Is (-1) if F has been used, otherwise zero.
A% Correct answer.
G$ Guess.
R% Questions right so far.
Q$ Addition or subtraction factor.

PROGRAM DESCRIPTION

Line Description

70 Accesses error routine.
80 Selects mode, removes cursor.
90 Accesses initialising procedure.
100 Accesses set up procedure.
120 Accesses question routine.
130 Adds 1 to T%. Accesses PROCEND if 10 questions asked, otherwise goes to line 120.
1190 Error routine for ERR <> Escape.
1200 Mode 7, no cursor.
1210-1230 Asks if the program is to be re-run.

PROCINIT

170 Defines N%. Dimensions Q%(11) and allows maximum space for G$.
180 Takes auto repeat off keyboard.
190 Chooses colours.
200 Defines envelope for sound signal.
210 CHR$224 goat stands.
220 CHR$225 goat jumps.
230 CRH$226 goat eats.
240 CHR$227 river and stepping stone.
250 CHR$228 left bank.
260 CHR$229 right bank. The two banks can differ if preferred.

PROCSETUP

290 Intialises R% and T%.
300 Prints left bank.
310 Prints river.
320 Prints right bank.
330 Delay and beep.
340 Prints goat.
350 Puts questions into 'pack'.

PROCPIC (X%,Y%,S%)
Draws picture, S%, at X%, Y%+20 in the required colour.

PROCBLANK
Blanks the lat goat.

PROCDELAY
Waits for 0/50 centiseconds.

PROCLEAR
Wipes the screen from Y1 to Y2.

PROCPLAY

590 Empties the keyboard buffer.
600 Chooses F from pack.
610 Takes F from pack.
620 Defines Q$ as '+' or '-' making sure that A% is always positive.
630 Prints question, input guess G$.
640 Clears and goes to 630 if G$ does not include a number.
650 Accesses PROCRIGHT or PROCWRONG.

PROCRIGHT

690 Sound 'pins'.
700-710 Clears question.
720 Blanks old goat.
730-740 Prints jumping goat.
750 Prints new goat.
760 Adds 1 to R%.

PROCWRONG

800 Buzz
810 Changes colour.
820-830 Prints right answer.
840 Clears sums.
850 Resets colour.

PROCEND

890 If R%(10, skips to line 970.
900-950 Goat eats grass.
960 Increases base N%.
970 Delays.

PROCTITLE

1010 Prints SUMS.
1020 Prints instructions.
1030-1160 Prints goat sequence for first run only.

Some of the variables are defined as integers to save variable storage space. The program structure makes it possible to

alter or omit the graphics without altering the main program. The program is made crashproof by taking the auto repeat off the keyboard to make it more difficult to fill the keyboard buffer with a silly answer. If the child succeeds in this, the input is well clear of the display, and no scrolling takes place. The input is trapped as a non-numeric or wrong answer.

In conclusion, the program can be made more versatile by making the base number and the number added the subject of an input routine.

BIRTHDAY

This program draws a picture of a birthday cake, plays 'Happy Birthday' and congratulates Christine on her fifth birthday.

VARIABLES

NAMES$	Name.
AGE$	Age.
C	Loop Variable.
A(X)	Pitch of musical note read from data.
B(X)	Duration of musical note read from data.

PROGRAM DESCRIPTION

Line Description

90 Selects mode, removes cursor.
100 Calls intialising procedure.
110 Calls name procedure.
120 Calls drawing procedure.
130 Calls music procedure.
140 Calls procedure to display age.

PROCNAME

180 Defines NAME$ and AGE$.

PROCINIT

220 Dimensions array for pitch and duration of the notes of the tune.
230 Defines a 'block' to use in the picture.
240 Defines a 'triangle' to use in the picture.
250 Changes Colour 1 to CYAN for the cake.
260 Changes Colour 2 to flashing BLUE and YELLOW for the candle flames.

PROCCAKE

300 Draws plate.
310 Draws cake.
320 Draws candles.
330 and 340 Draw candle flames.
350 Prints 'HAPPY BIRTH-DAY'.
360 Centres the name and prints it.

PROCMUSIC

400 Reads 25 values of A(X) and B(X).
410 Plays the 25 notes. If a note has the same pitch as the next one, the program will pause between notes.

PROCAGE

470 Displays the age up to 101 times on the screen at random.

The DATA is the pitch and the duration of each note of 'Happy Birthday' in the key of C in moderator.

In Conclusion, this program can be altered fairly easily. PROCDRAW could be changed to draw a better picture, PROCAGE ommitted on request, and PROCNAMES customised using a crash input routine.

RAINBOW

This is a race between two children to see who can build a coloured tower first. After the title is displayed, the children are invited to input their names. They are told to roll a colour dice, by pressing any letter on the keyboard, when they see their own name on the screen.

They build a section of their tower when the colour on their dice matches the target colour on the middle tower.

The screen clears, the central tower is shown, and the dice is put on the left-hand side for the first player. If the dice shows RED after play, the player can start to build the tower. The dice is transferred to the right hand side for the other player.

When a player has built all six sections, the computer plays a tune, and congratulates the winner before starting again. To change the players names, ESCAPE, and answer Y to a re-run.

VARIABLES

A$	Name of player who wins the toss.
B$	Name of other player.
SA	Sections completed by A.
SB	Sections completed by B.
TA	Next target for A.
TB	Next target for B.
O$	Input for re-run.
T$	Title string.
N,V	Position of letter in word.
X(V)	ASCII of letter in position V.
W$	Underline string.
M	Number of changes of colour in title sequence.
T	Colour of central tower section.
A$(1)	First name entered.
A$(2)	Second name entered.
G	Number of turns.
P	Position of dice.
V$	Letter pressed to roll dice.
C	Loop variable in dice procedure.
X	Dice colour.
WAIT	Delay time.
PL	Player number.

PROGRAM DESCRIPTION

Line Description

90 Accesses error routine.
100 MODE 2, no cursor.
110 Accesses initial routine.
120 Displays title.
130 Accesses name input routine and instructions.
140 Displays title.
150 Builds central tower.
160 Prints current player's name.
170 Prints dice.
180 Dice roll.
190 Adjusts score.
200 Goes to line 160 if the game is unfinished, otherwise win procedure.
210 Tosses for first go on next game.
220 Waits for five seconds.
230 Initialises scores, targets, turns, and dice position.
240 Clears screen and goes to line 140.
260 Reports error, if not Escape key.
270-300 Ask if program is to be re-run.

PROCTITLE

This is a general procedure to print T$ underlined by W$ in multicolours.

330 Defines T$.
340-360 Put ASCII of letter in position V in array.
370-390 Print title.
400 Defines W$.
410-420 Print underline.

PROCTOWER

460-490 Print central tower two rows of six blocks of each colour.

PROCNAMES

530 Blanks old name.
540 Prints A$ after even number of turns, otherwise B$.

PRODICE

580 Defines position of dice.
590 Prints dice.

PROCPLAY

650 Empties buffer.
660 Waits for key to be pressed.
670-720 Roll dice and put colour in centre.
730 Waits two seconds.
740 Blanks dice.

PROCWIN

790 If A wins, puts turns to two, else one.
800 Prints name.
810 Colour 7 to 8.
820 Prints WINS.
830 Plays tune, twice.
840 Resets Colour 7.

PROCHAMP

This is just a short phrase, and an array is not used.

CONTINUED OVER

890-940 Play 'We are the champions' pausing between each note.

PROCP

970 Pause.

PROCINST

1010-1080 Name input with checking.

1090 Chooses who will start, otherwise the first player will have an unfair advantage.

1100-1130 Show instructions.

PROCDELAY

1170 Pauses for WAIT centiseconds.

PROCINIT

1210 Sets up user defined block.

1220 Puts totals to zero.

1230 Dimensions array.

1240 No auto repeat.

PROSCORE

1280 Adds one to turns.

1290 Goes to line 1320 if A is playing.

1300 ENDPROC if not target colour.

1310 Adds one to TB and SB print section of tower.

1320 ENDPROC if not target colour.

1330 Adds one to TA and SA print section of tower.

PROCTOSS

1370-1380 Select which player will start.

It is necessary to put PROCTOSS before PROCDELAY because setting TIME during a delay can bias the pseudo random number generator if enought time has not elapsed in between.

The program can be adapted for more players. It should also be possible to convert this program for use on a Model A in MODE 7, putting graphics control codes at the start of some of the lines. The title and instructions display would also need adjusting.

SIMPLE GRAPHICS: SUMS

```
  10 REM***************
  20 REM*             *
  30 REM*   SUMS BY   *
  40 REM*  M.STANGER  *
  50 REM*             *
  60 REM***************
  70 ON ERROR GOTO1190
  80 MODE 5:!&FE00=&10200A
  90 PROCINIT
 100 PROCSETUP
 110 PROCTITLE:PROCSETUP
 120 PROCPLAY
 130 T%=T%+1 :IF T%=10 PROCEND:GOTO100 ELSE PRINTTAB(4+T%,24);T%:GOTO1
20
 140 END
 150 :
 160 DEF PROCINIT
 170 N%=2:DIM0%(11):G$=STRING$(255," ")
 180 *FX11,0
 190 VDU 19,1,2,0,0,0,19,2,4,0,0,0
 200 ENVELOPE1,1,0,0,0,1,1,1,126,0,-10,-3,126,126
 210 VDU 23,224,&06,&83,&FF,&7F,&79,&48,&48,&48
 220 VDU 23,225,&06,&83,&FF,&7F,&79,&4C,&82,&80
 230 VDU 23,226,&00,&80,&F3,&79,&7F,&8F,&88,&89
 240 VDU 23,227,&7C,&44,&44,&7C,&FF,&FF,&FF,&FF
 250 VDU 23,228,&FF,&FF,&FF,&FF,&FF,&FF,&FF,&FF
 260 VDU 23,229,&FF,&FF,&FF,&FF,&FF,&FF,&FF,&FF
 270 ENDPROC
 280 DEFPROCSETUP
 290 R%=0:T%=0:CLS
 300 FOR I%=1 TO 4:PROCPIC(I%,1,4):NEXT
 310 FOR I%=5 TO 13:PROCPIC(I%,1,3):NEXT
 320 FOR I%=14 TO 18:PROCPIC(I%,1,5):NEXT
 330 PROCDELAY(2000):PRINTCHR$7
 340 PROCPIC(4,0,0)
 350 FOR I%=0 TO 11:0%(I%)=0:NEXT
 360 ENDPROC
 370 :
 380 DEFPROCPIC(X%,Y%,S%)
 390 IF S%>3 COLOUR1:GOTO410
 400 IF S%=3 COLOUR2 ELSE COLOUR3
 410 PRINTTAB(X%,Y%+20);CHR$(224+S%);
 420 ENDPROC
 430 :
 440 DEFPROCBLANK(X%)
 450 PRINTTAB(X%,20);" ";
 460 ENDPROC
 470 :
 480 DEFPROCDELAY(Q)
 490 TIME=0:REPEAT UNTIL TIME>Q/50
 500 ENDPROC
 510 :
 520 DEFPROCCLEAR(I1,I2)
 530 FOR I%=I1 TO I2
 540   PRINTTAB(0,I%);STRING$(20," ")
 550   NEXT
 560 ENDPROC
 570 :
 580 DEFPROCPLAY
 590 *FX 15,0
 600 F=RND(10):IF Q%(F)=-1 GOTO600
 610 Q%(F)=-1
 620 IF RND(2)=1 AND N%>F Q$=" - ":A%=N%-F ELSE Q$=" + ":A%=N%+F
 630 PRINTTAB(5,7);N%;Q$;F;" = ":INPUTTAB(13,7)G$
 640 IF VAL(G$)=0 PROCCLEAR(7,10):GOTO630
 650 IF VAL(G$)=A% PROCRIGHT ELSE PROCWRONG
 660 ENDPROC
 670 :
 680 DEFPROCRIGHT
 690 SOUND1,1,180,3
 700 PROCDELAY(2000)
 710 PROCCLEAR(7,10)
 720 PROCBLANK(R%+4)
 730 PROCPIC(R%+5,0,1)
 740 PROCDELAY(4000)
 750 PROCPIC(R%+5,0,0)
 760 R%=R%+1
 770 ENDPROC
 780 :
 790 DEFPROCWRONG
 800 SOUND0,-15,6,10
 810 VDU 19,2,11,0,0,0:COLOUR 2
 820 PRINTTAB(5,12);N%;Q$;F;" = ";A%
 830 PROCDELAY(20000)
 840 PROCCLEAR(7,19)
 850 VDU 19,2,4,0,0,0:COLOUR 3
 860 ENDPROC
 870 :
 880 DEFPROCEND
 890 IF R%<10 GOTO970
 900 FOR C=1 TO 6
 910   SOUND0,-10,2,1
 920   PROCDELAY(1000)
 930   PROCPIC(T%+4,0,2)
 940   PROCDELAY(2000)
 950   NEXT
 960 N%=N%+1
 970 PROCDELAY(20000)
 980 ENDPROC
 990 :
1000 DEFPROCTITLE
1010 COLOUR 2:PRINTTAB(5,2);"S U M S"
1020 COLOUR 1:PRINT:PRINT:PRINT:PRINT"The goat can cross"'"the river
if your"'"sums are right":COLOUR 3
1030 IF N%>2 PROCDELAY(15000):ENDPROC
1040 FOR C%=0 TO 9
1050   PROCDELAY(2000)
1060   PROCBLANK(C%+4)
1070   PROCPIC(C%+5,0,1):SOUND1,1,180,3
1080   PROCDELAY(4000)
1090   PROCPIC(C%+5,0,0)
1100   NEXT
1110 FOR C=1 TO 6
1120   SOUND0,-10,2,1
1130   PROCDELAY(1000)
1140   PROCPIC(C%+4,0,2)
1150   PROCDELAY(1000)
1160   NEXT
1170 ENDPROC
1180 :
1190 IF ERR<>17 REPORT:PRINT" at line ";ERL:END
1200 MODE 7:!&FE00=&10200A
1210 PRINTTAB(1,10);"DO YOU WANT ANOTHER GO(Y/N)?":G$=GET$
1220 IF G$="Y"  RUN
1230 IF G$="N" PRINT:PRINT"       GOODBYE" :END ELSE PROCCLEAR(10,11):
GOTO1210
```

SIMPLE GRAPHICS: BIRTHDAY

```
>LIST
  10 REM***************
  20 REM*             *
  30 REM*  BIRTHDAY   *
  40 REM*     BY      *
  50 REM*  M.STANGER  *
  60 REM*             *
  70 REM***************
  80 :
  90 MODE5:VDU 23;8202;0;0;
```

```
100 PROCINIT
110 PROCNAME
120 PROCCAKE
130 PROCMUSIC
140 PROCAGE
150 END
160 :
170 DEFPROCNAME
180 NAME$="CHRISTINE":AGE$=" 5 "
190 ENDPROC
200 :
210 DEFPROCINIT
220 DIM A(50):DIM B(50)
230 VDU 23,224,&FF,&FF,&FF,&FF,&FF,&FF,&FF,&FF
240 VDU 23,225,1,3,7,15,31,63,127,255
250 VDU19,1,6,0,0,0
260 VDU19,2,11,0,0,0
270 ENDPROC
280 :
290 DEFPROCCAKE
300 PRINTTAB(0,30);STRING$(19,CHR$224)
310 COLOUR 1:FOR C=24 TO 29:PRINTTAB(2,C);STRING$(15,CHR$224):NEXT
320 COLOUR 3:FOR C=18 TO 23:PRINTTAB(4,C);" ";CHR$224;"    ";CHR$224;"
";CHR$224:NEXT
330 COLOUR 2:PRINTTAB(4,17);CHR$225;CHR$224;"  ";CHR$225;CHR$224;"  ";
CHR$225;CHR$224
340 COLOUR 2:PRINTTAB(5,16);CHR$225;"    ";CHR$225;"    ";CHR$225
350 PRINTTAB(2,5)"HAPPY  BIRTHDAY"
360 PRINTTAB(10-(LEN(NAME$)/2),9);NAME$
370 ENDPROC
380 :
390 DEFPROCMUSIC
400 FOR X=1 TO 25:READ A(X):READB(X):NEXT
410 FOR X=1 TO 25
420   SOUND1,-15,A(X),B(X):IF A(X)=A(X+1) SOUND1,0,0,2
430   NEXT
440 ENDPROC
450 :
460 DEFPROCAGE
470 CLS:FOR X=0 TO 100:COLOUR RND(3):PRINTTAB(RND(15),RND(30));AGE$:NE
XT
480 ENDPROC
490 :
500 DATA 53,4,53,4,61,8,53,8,73,8,69,16
510 DATA 53,4,53,4,61,8,53,8,81,8,73,16
520 DATA 53,4,53,4,101,8,89,8,73,8,69,8
530 DATA 61,8,93,4,93,4,89,8,73,8,81,8,73,16
```

SIMPLE GRAPHICS: RAINBOW

```
 10 REM**************
 20 REM*            *
 30 REM*  RAINBOW   *
 40 REM*     BY     *
 50 REM* M.STANGER *
 60 REM*            *
 70 REM**************
 80 :
 90 ON ERROR GOTO270
100 MODE 2:!&FE00=&10200A
110 PROCINIT
120 PROCTITLE
130 PROCINST
140 PROCTITLE
150 PROCTOWER
160 PROCNAMES
170 PROCDICE
180 PROCPLAY
190 PROCSCORE
200 IF SA=6 OR SB=6 PROCWIN ELSEGOTO160
210 PROCTOSS
220 PROCDELAY(500)
230 G=0:SA=0:SB=0:TA=1:TB=1:P=0
240 CLS:GOTO140
250 :
260 IF ERR<>17 REPORT:PRINT" AT LINE ";ERL:END
270 PRINTTAB(0,3)"ANOTHER GAME(Y/N)":O$=GET$
280 IF O$="Y" RUN
290 IF O$="N" CHAIN""ELSE GOTO270
300 GOTO270
310 :
320 DEFPROCTITLE
330 T$="RAINBOW"
340 FOR V=1 TO 7
350   C$=MID$(T$,V,1)
360   X(V)=ASC(C$):NEXTV
370 FOR M=1 TO 10
380   FOR N=1 TO 7
390     PRINTTAB(1+2*N,1);:COLOUR RND(6):PRINTCHR$(X(N))
400     W$="**"
410     PRINTTAB(1+2*N,2);:COLOUR RND(6):PRINTW$
420     NEXTN:PRINTTAB(16,2);" ";:NEXTM
430 ENDPROC
440 :
450 DEFPROCTOWER
460 FOR T=1 TO 6
470   PRINTTAB(6,20-2*T);:COLOUR T:PRINTSTRING$(7,CHR$224)
480   PRINTTAB(6,19-2*T);STRING$(7,CHR$224)
490   NEXTT
500 ENDPROC
510 :
520 DEFPROCNAMES
530 COLOUR 7: PRINTTAB(0,21);STRING$(20," ")
540 IF G MOD 2=0 PRINTTAB(5-LEN(A$)/2,21);A$ ELSE PRINTTAB(15-LEN(B$)/
2,21);B$
550 ENDPROC
560 :
570 DEFPROCDICE
580 IF G MOD 2=0 P=2 ELSE P=12
590 FOR C=1 TO 6
600   PRINTTAB(P,23+C);STRING$(6,CHR$224)
610   NEXTC
620 ENDPROC
630 :
640 DEFPROCPLAY
650 *FX 15,1
660 V$=GET$:IF V$="" GOTO660
670 FOR C=1 TO 10
680   X=RND(6)
690   PRINTTAB(P+2,26);:COLOUR X:PRINTCHR$224;CHR$224
700   PRINTTAB(P+2,27);:COLOUR X:PRINTCHR$224;CHR$224
710   SOUND0,-15,3,1:SOUND0,0,0,1
720   NEXTC
730 PROCDELAY(200)
740 FOR C=1 TO 6
750   PRINTTAB(0,23+C);STRING$(18," ");:NEXTC
760  ENDPROC
770 :
780 DEFPROCWIN
790 IF TA>TB LET G=2 ELSE G=1
800 PROCNAMES
810 VDU19,7,8,0,0,0
820 IF G=2 PRINTTAB(3,22);"WINS" ELSE PRINTTAB(13,22);"WINS"
830 PROCCHAMP:PROCCHAMP
840 VDU19,7,7,0,0,0
850 ENDPROC
860 :
870 DEFPROCCHAMP
880 FOR P=1 TO 2
890   SOUND1,-15,81,4:PROCP:NEXT
900 SOUND1,-15,69,2:PROCP
910 SOUND1,-15,89,6:PROCP
920 SOUND1,-15,81,4
930 FORP=1 TO5:PROCP:NEXT
940 ENDPROC
950 :
960 DEFPROCP
970 SOUND1,0,0,1
980 ENDPROC
990 :
1000 DEFPROCINST
1010 FORPL=1 TO 2
1020   COLOURPL:PRINTTAB(0,4*PL)"NAME,PLEASE,PLAYER ";PL;
1030   INPUTTAB(5,4*PL+2)A$(PL)
1040   IF A$(PL)="" GOTO1020
1050   IF LEN(A$(PL))>10 CLS: PRINTTAB(0,4*PL);"A SHORT NICKNAME,PLEASE
,PLAYER ";PL;:GOTO1030
1060   IF A$(PL)=""GOTO1020
1070   VDU7
1080   NEXT
1090 PROCTOSS
1100 COLOUR3:PRINT:PRINT"When you see your""'""name,it is your turn"
1110 PRINT"to shake the dice""'""by pressing a letter"
1120 COLOUR6:PRINT"You start building""'""your tower when""'""the colour
on your""'""dice is the same""'""as the first colour""'""on the big tower"
1130 PROCDELAY(1000):CLS
1140 ENDPROC
1150 :
1160 DEFPROCDELAY(WAIT)
1170 NOW=TIME:REPEAT:UNTIL TIME>NOW+WAIT
1180 ENDPROC
1190 :
1200 DEFPROCINIT
1210 VDU 23,224,255,255,255,255,255,255,255,255
1220 G=0:SA=0:SB=0:TA=1:TB=1:P=0
1230 DIMX(7):DIMA$(2)
1240 *FX11,0
1250 ENDPROC
1260 :
1270 DEFPROCSCORE
1280 G=G+1
1290 IF G MOD 2=1 GOTO1320
1300   IF X<>TB ENDPROC
1310 TB=TB+1:SB=SB+1:PRINTTAB(15,22-2*TB);:COLOUR X:PRINTSTRING$(4,CHR$
224) :PRINTTAB(15,21-2*TB);STRING$(4,CHR$224) :PRINTCHR$7:ENDPROC
1320 IF X<>TA ENDPROC
1330   TA=TA+1:SA=SA+1:PRINTTAB(0,22-2*TA);:COLOUR X:PRINTSTRING$(4,CHR$
224):PRINTTAB(0,21-2*TA);STRING$(4,CHR$224):PRINTCHR$7
1340 ENDPROC
1350 :
1360 DEFPROCTOSS
1370 IF RND(1)>.5 A$=A$(1):B$=A$(2):ENDPROC
1380 A$=A$(2):B$=A$(1):ENDPROC
```

Garbage Collecting

H Roberts

Some helpful hints here show you how to make the most of memory capacity using space saving techniques and artful ways of string handling.

Most Microsoft BASICS have a large area of contiguous memory used for strings. This space is reserved at the beginning of the program with the CLEAR statement, and filled up from the far end as strings are defined. The names of the strings, their lengths and pointers to their starts, are stored in the normal variable area at the start of memory, after the program.

When the string space at the end of memory is full, the interpreter goes through a process called 'garbage collection', where it looks through the string area for strings that are not pointed to and clears them out, while at the same time compressing the rest of the strings. If a large area of space has been allocated and a lot of strings are stored in it, this garbage collection can take several seconds and has to be done frequently so it is cumbersome and slow.

BETTER READ

Another feature of many BASICS is the way they read strings in from data. Instead of moving the string into the string space at the end of memory, they simply set up the variable pointing to the data statement, not using any string space at all. This is very useful in (for example) adventure games where a large number of strings have to be fitted into memory. When these strings are read in, they take no more space than an equal number of numeric variables.

The graphics modes of the BBC Micro prevent this system being used. There are seven graphics modes using either 1K, 8K, 10K, 16K or 20K of memory. This variable amount of memory means that the top of memory (HIMEM) alters whenever the graphics mode is changed. If the strings were stored at the end of memory, immediately before the graphics area, then they would have to be moved and their pointers adjusted every time the graphics mode was changed. This would be impractical, so a completely different system is used.

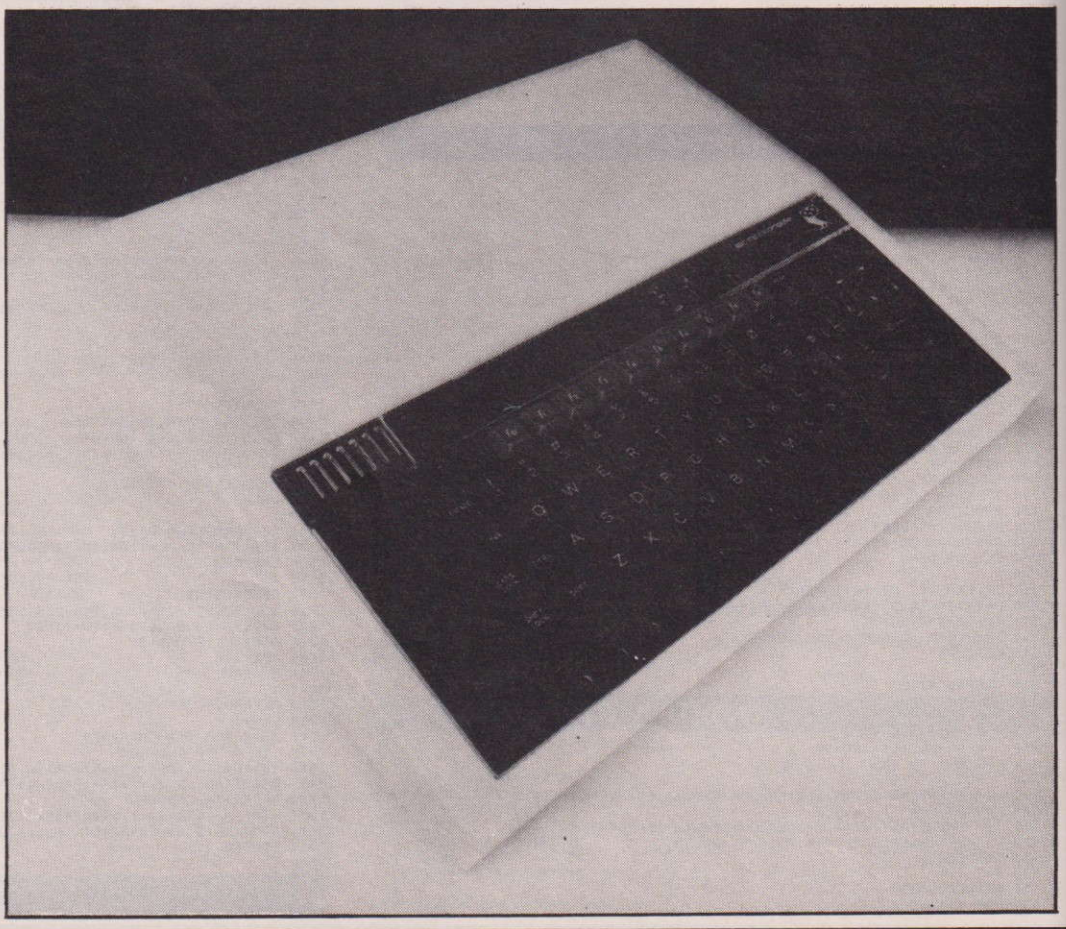
String variables on the BBC Micro are stored at the beginning of memory with the other variables. When a string variable is first defined, the string itself is stored immediately after its pointers. If the string is made longer than it was originally, it is moved to a completely new place at the end of the variable storage area, taking up much more space. What's more, if it is made longer again, it will be moved again taking up even more space.

When the variable space is filled, BBC BASIC will not garbage collect. The redundant strings will remain in their original places and the memory they take up cannot be used for anything else.

Strings read from data are handled no differently from any others. The string is copied directly into the variable data area, and therefore takes up twice as much space as on a normal Microsoft BASIC.

TROUBLE?

This unusual method of storing strings can cause a great deal of



trouble to the programmer, it means you have to be extra careful when handling strings and a number of guidelines would be very useful.

First, don't read a large number of strings in from data because they will be stored twice and take up twice the amount of memory. There are several ways to avoid this problem, depending on the application.

1. In many cases it is only necessary to read the strings in one at a time; if this is the case, a computed restore is the best method for reading the strings in. Put the strings in consecutive data statements a fixed distance apart (for example, 1000, 1010, 1020 . . .). The line number in which the Ith string is stored is then easily calculated (in the example, the Ith string is stored at 990+1*10). The program can then restore to that line number and read the required string (RESTORE 990+1*10:READ NAME$).
2. The strings to be read in could be stored in a cassette file recorded immediately after the program and read out of the file straight into an array. No data statements are needed for this method and the strings are thus only stored in memory once.

Second, don't change the lengths of strings after they have been first defined since they will be moved and leave a large amount of redundant memory. This problem can be overcome by initialising all strings to their maximum length. So, if you were storing a large string array of names and addresses, you should first set all the strings in the array to a maximum length of about say, twenty spaces. You should then ensure that none of the input strings exceeds twenty spaces in length by inputing the string into another variable and checking the length before inserting it into the string array. For example:

```
  10  DIM NAME$(200)
  20  FOR I=1 TO 200:NAME$(I)=STRING$(20," "):NEXT I
   :                  :
   :                  :
1100  INPUT NAME$
1110  IF LEN(NAME$)>20 THEN PRINT "Name exceeds maximum
      length.":GOTO 1100
1120  NAME$(J)=NAME$
```

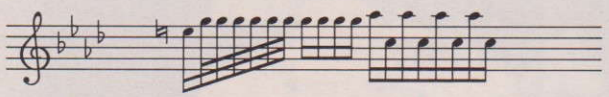
Third, reuse variables. Most professional software houses deplore this method of saving space because it makes programs less readable and less maintainable. However, they do not have the same problem with space as micro programmers. String variables especially use up a lot of space, so reuse these as much as possible.

In conclusion, use these methods only when necessary. A 200 byte saving on space is not worth it if it makes the program unreadable. Above all program readability is essential, and space is only important if the program is too large to fit into memory.

Very Sound Advice

Try these routines and hear your micro play Vivaldi.

Have you ever tried to use the SOUND command to produce the same note played rapidly several times in succession? A particularly challenging example is the following bar taken from the lead violin part in Vivaldi's 'The Four Seasons':



This should be played smoothly, and the faster notes played at a rate of about ten per second. Restricting ourselves to Channel 1 only, and bearing in mind that the duration parameter is in units of one-twentieth of a second (ie SOUND X, Y, Z, 20 plays for one second). Our first attempt might be as shown in Listing 1, where we transcribe the notes quite literally and assign the semiquavers (♪) a time value of four and the demisemiquavers (♪) a time value of two. However, this does not work as no spaces at all are left between the notes, and the second to eleventh short notes are sounded as one long one.

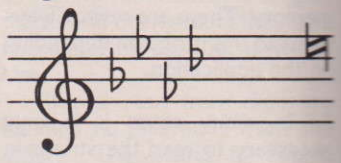
POINTS TO NOTE

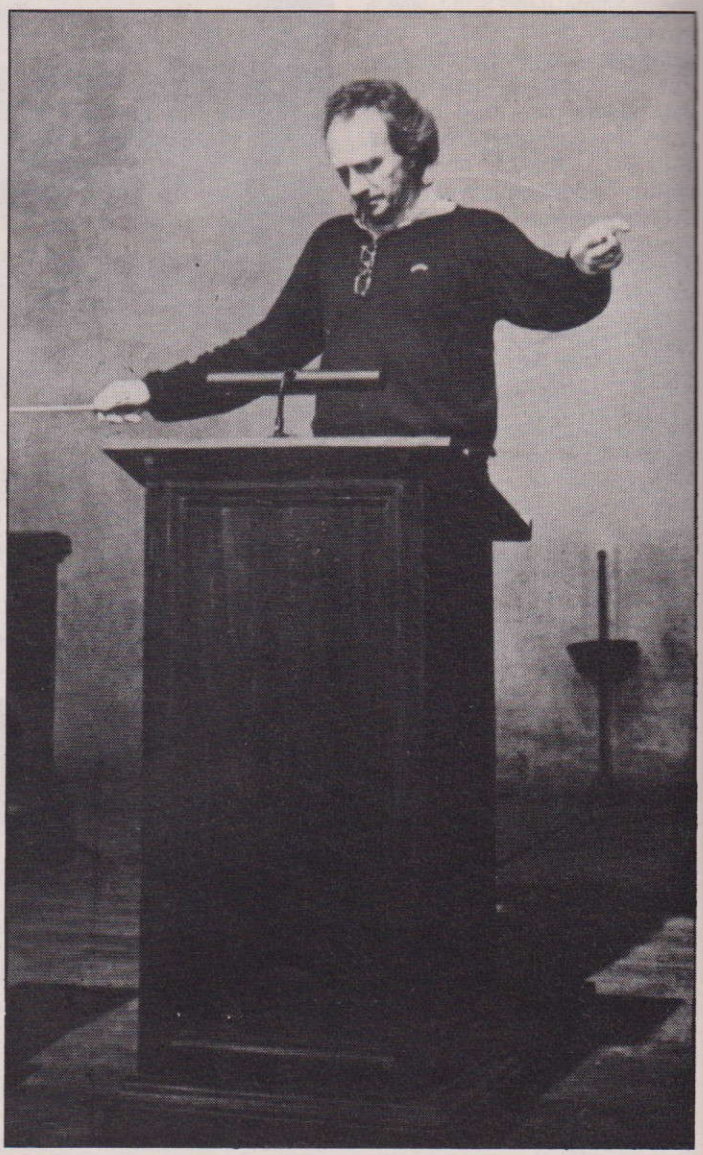
The next step is to insert between each note a SOUND command which plays at zero volume for one-twentieth of a second (ie SOUND 1, 0, 0, 1) and assign time values of three to ♪ and one to ♪ in order to keep the timing correct, and this is shown is Listing 2. This plays the music, of course, but doesn't it sound harsh and staccato? I certainly would not accept it as being remotely as Vivaldi intended.

One way round this is to use a fact that the BBC **User Guide** omits, and that is a duration parameter of zero gives rise to a note lasting for 1/200 of a second which is ideal for making very short pauses. Now we try assigning time values of four to ♪ and two to ♪ , and use zero for all spaces between each note as is shown in Listing 3. The music is now played in a reasonably acceptable fashion—certainly far preferable to the other two attempts. There is, however, one point to beware of when using this idea in two or three-part harmony—these small gaps may build up and destroy the togetherness of the three channels, and it is advisable to synchronise at the beginning of the next bar. For example, to play all three channels together after the above bar, one might use:

```
SOUND 513,V1,P1,T1:
SOUND 514,V2,P2,T2:
SOUND 515,V3,P3,T3
```

It must, however, be admitted that every one of the above methods has some limitations, and the program in Listing 3 suffers from needing two SOUND commands for each note, and also from a slight distortion in the timing. Surely there is a more elegant and accurate way of solving this problem. The ENVELOPE command may be used to control the way a note behaves as a function of time, and we turn to it now for an alternative solution. The note must start sounding as quickly as possible, keep sounding for an appropriate length of time and then decay to silence, and the whole process must take place in just under a tenth of a second for ♪ , and a fifth of a second for ♪ . A different ENVELOPE command is needed for each note-length, and Listing 4 shows how it may be done. Provided you don't mind sacrificing some of your ENVELOPE commands to do the job, the method is ideal for this type of problem or for generating staccato notes, but requires just a little more thought for general use.





B Landsberg

PROGRAM LISTING

Listing 1.

```
10 REM ** 1ST ATTEMPT TO PLAY MUSIC  **
20 REM **    DOESN'T WORK AT ALL      **
30 SOUND 1,-15,116,4
40 FOR I = 1 TO 6
50 SOUND 1,-15,128,2
60 NEXT
70 FOR I = 1 TO 4
80 SOUND 1,-15,128,4
90 NEXT
100 FOR I = 1 TO 4
110 SOUND 1,-15,132,4
120 SOUND 1,-15,100,4
130 NEXT
```

Listing 2.

```
10 REM ** 2ND ATTEMPT TO PLAY MUSIC  **
20 REM ** BETTER - BUT STILL BAD     **
30 SOUND 1,-15,116,3: SOUND 1,0,0,1
40 FOR I = 1 TO 6
50 SOUND 1,-15,128,1: SOUND 1,0,0,1
60 NEXT
70 FOR I = 1 TO 4
80 SOUND 1,-15,128,3: SOUND 1,0,0,1
90 NEXT
100 FOR I = 1 TO 4
110 SOUND 1,-15,132,3: SOUND 1,0,0,1
120 SOUND 1,-15,100,3: SOUND 1,0,0,1
130 NEXT
```

Listing 3.

```
10 REM ** 3RD ATTEMPT TO PLAY MUSIC  **
20 REM ** FAR BETTER THAN THE OTHERS **
30 SOUND 1,-15,116,4: SOUND 1,0,0,0
40 FOR I = 1 TO 6
50 SOUND 1,-15,128,2: SOUND 1,0,0,0
60 NEXT
70 FOR I = 1 TO 4
80 SOUND 1,-15,128,4: SOUND 1,0,0,0
90 NEXT
100 FOR I = 1 TO 4
110 SOUND 1,-15,132,4: SOUND 1,0,0,0
120 SOUND 1,-15,100,4: SOUND 1,0,0,0
130 NEXT
```

Listing 4.

```
10 REM **      MORE ELEGANT APPROACH  **
20 REM ** USES ENVELOPE TO MAKE GAPS **
24 ENVELOPE 1,1,0,0,0,0,0,0,126,-1,-126,0,126,119
28 ENVELOPE 2,2,0,0,0,0,0,0,126,-1,-126,0,126,119
30 SOUND 1,2,116,4
40 FOR I = 1 TO 6
50 SOUND 1,1,128,2
60 NEXT
70 FOR I = 1 TO 4
80 SOUND 1,2,128,4
90 NEXT
100 FOR I = 1 TO 4
110 SOUND 1,2,132,4
120 SOUND 1,2,100,4
130 NEXT
```

Symbolic Disassembler

Is assembly code double Dutch to you? If so, try our disassembling utility!

The BBC Micro is a remarkable machine, not least because of the built-in assembler which allows programmers to use fast 6502 machine-code in BASIC programs. The machine code may be entered using labels and mnemonics which indicate the purpose of each instruction. The assembler converts the text into binary patterns which the 6502 can recognise.

This symbolic disassembler performs the opposite function to that of the built-in assembler. It converts the contents of RAM or ROM into mnemonics and labels which may be displayed or printed for later reference. Mnemonics are symbolic names such as 'BRK' meaning BReaK or 'LDA', short for LoaD Accumulator.

The revised **User Guide** contains a short summary which shows how the assembler is used. The manual also explains how a few of the operating system routines may be used from a machine code program. However, it contains no details of the BASIC ROM and the other languages and utilities planned.

THE HIDDEN MEANING

The disassembler allows a programmer to look through the 15,000 lines of machine code in the BBC Micro's ROM. It indicates the instructions and data areas which the programmers at Acorn used. The ROM instructions are displayed in the form of standard 6502 assembler mnemonics.

Whenever one of the 'documented' operating system addresses is referenced, the symbolic disassembler replaces the numeric address with the label given by Acorn. For example, a fetch from address &0201 Hex would consult the most significant byte of the user vector Its label would be USERV+1. Likewise, a subroutine call to write a 'New line' sequence would be displayed as JSR OSNEWL. A full table of the symbols used is in an appendix at the end of the revised BBC **User Guide.**

The target address of relative branch instructions is automatically converted from two's complement binary form into a 16-bit address, making the destination of every branch obvious. If that 16-bit address corresponds to a label (as at the OSASCI entry point) then the label is printed rather than the Hex address, just as for jumps and subroutine calls.

ROM RUBBISH

One of the flaws of an automatic program to convert data into mnemonics is that it cannot tell the difference between data (text, perhaps) and machine instructions. The byte 32 could represent either a space character (in ASCII code), the hexadecimal value, &20, or the machine code instruction, JSR. The only certain way to find out what every byte represents is to execute the entire ROM (including loops) from every start address and that would take years, even at machine code speeds.

Faced with this problem, the disassembler does not attempt to differentiate between data and program in the ROM. Luckily, it is possible to make the purpose of most bytes clear simply by disassembling each location in numeric, mnemonic and textual form. The format of the disassembly is shown below:

Address	hex data
&6012	20 FF 33
&6013	00

If you disassemble the BBC ROM from address &8000 Hex onwards you will find that the first few bytes contain jump instructions and other lines of machine code. The mnemonic column indicates the programming used.

The ASCII column is pretty meaningless at this point, since the machine code instructions have arbitrary values. Some bytes do not correspond to teletext characters. If these were printed, strange things might happen to the display (they could be control characters) and consequently, bytes with a value less than space (32) or greater than cursor (127) are displayed as full stops. Remove line 543 if you wish to disable this feature.

A little further on through the ROM you will find that the mnemonics become meaningless. Some of them will be shown as '???'. These are opcodes which the 6502 will not recognise as instructions. On early PET computers, some of these were given the mnemonic HCF, standing for Halt and Catch Fire, since they caused the 6502 to go wild if they were accidentally encountered. The processor would count repeatedly from 0 to 65535 on its address lines and ignore everything else. Luckily, later PETS and the BBC Micro do not suffer from this problem!

The meaningless instructions occur because you are disassembling an area of ROM which contains data rather than instructions. In fact this is obvious, since the ASCII column of the display will show that you are disassembling the BBC Micro copyright message. Later in the ROM, other 'nonsense' areas will be found. These may not even make sense as text, in which case they probably correspond to tables of addresses or values. In either case, the numbers used can be easily found by consulting the 'Hex data' column.

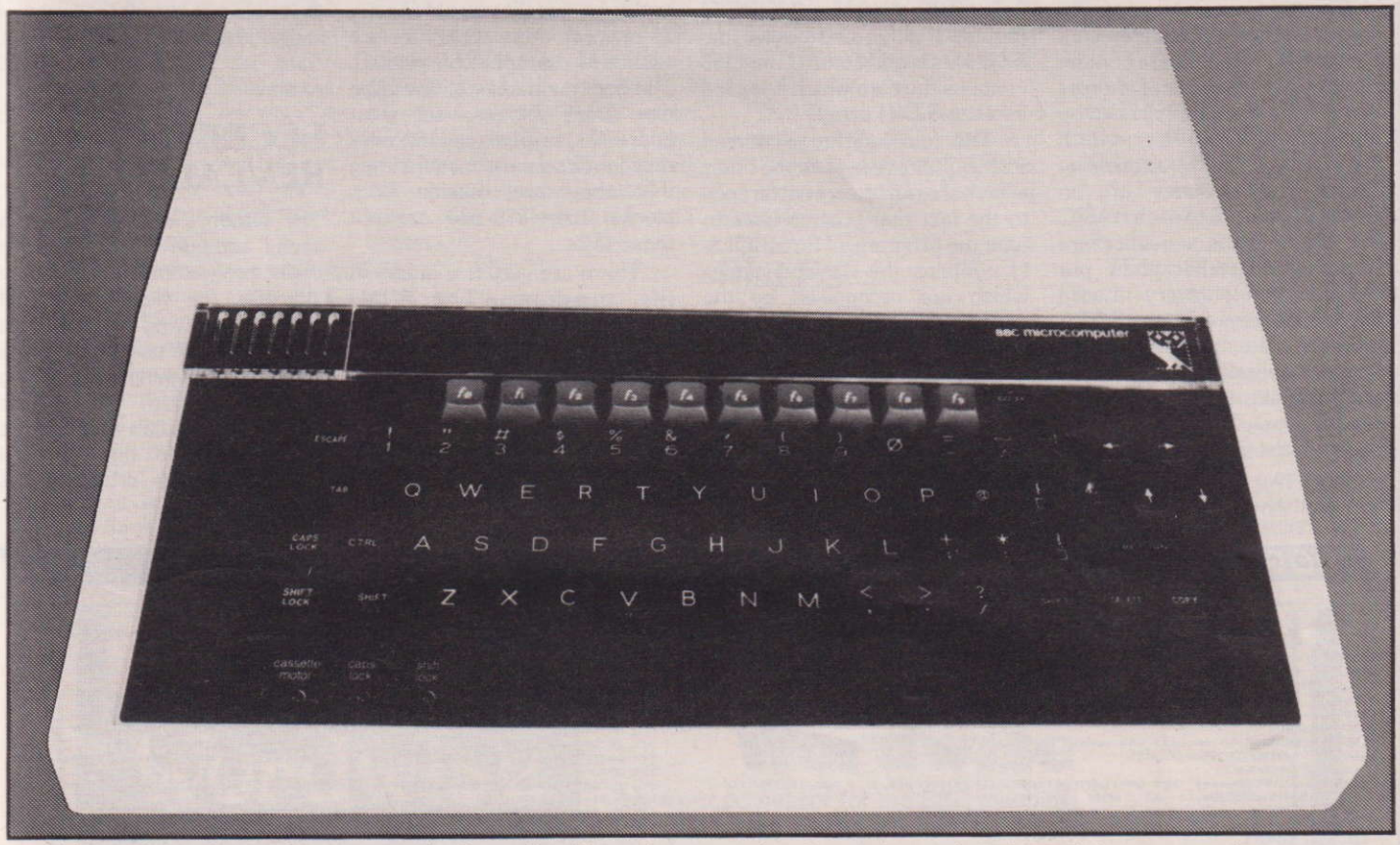
In a few parts of the BBC operating system you may encounter long strings of zero bytes. These are unused areas of the ROM which have been left unprogrammed. Acorn have deliberately left some areas free so that they have space to correct their mistakes (in later versions of the operating system) without overflowing onto a separate ROM. Unfortunately, these zeros are fixed when the ROM is manufactured, and consequently you cannot use those areas for your own purposes. They are disassembled as BRK instructions.

Sometimes data areas will force the disassembler out of step with the 6502 instructions. If the last byte of a data area was &20 Hex, for example, the next two bytes would be disassembled as an address (for the rest of a three byte JSR) rather than as instructions. After a few lines, the disassembler will automatically become synchronised and work correctly, but you may encounter a few lines of 6502 nonsense before the disassembler realises that the data has finished. You may correct the listing by disassembling from the address immediately after the end of the data. This problem does not occur often and it is usually quite obvious when it crops up.

THE PROGRAM

The program uses the Teletext display and consequently it will fit comfortably onto a 16K computer (Model A). Alter the value of PAGE before you load the disassembler if you wish to examine a program which normally loads into memory in the BASIC area (&0E00 Hex on a cassette system). Move PAGE to

S Goodwin



an address above the machine code, and then load the disassembler above it.

When first RUN, the disassembler will display a 'Please Wait' message while it sets up tables of opcodes and labels. After a few seconds the main display will appear, explaining how the program may be used. Type in the address where the program you want to disassemble is located. Prefix the address with an '&' if the address is in hexadecimal. Type Control B after the number if you wish the disassembly to be printed out (Control C will turn off printing).

Negative addresses or values greater than 65535 (&FFFF Hex) are not allowed. If you typed in a valid address the program will then begin to disassemble each line of the machine code in the manner explained earlier. To pause the listing at any point, press the Space Bar. If you then decide to stop the disassembly, press the Return key. You will then be asked to type in a new start address. Otherwise, press the Space Bar again and the program will resume where it left off.

HOW IT WORKS

The disassembler is quite a simple program. It revolves around the use of four tables, declared as arrays in line 150. The array OP$() contains the text associated with each opcode value. OP$(32) contains the text 'JSR', for instance, since the instruction numbered 32 is a subroutine jump and JSR is the appropriate 6502 mnemonic. Various elements contain '???' since they correspond to numbers which do not have legal 6502 opcodes.

The array, AD%(), contains a number for each opcode value. The number varies depending upon the format or 'addressing mode' of each instruction. Table 1 shows the relationship between the numbers and the 6502 addressing modes.

If the names and mnemonics in the table do not make much sense to you then you are probably not in need of a disassembler! Unfortunately, there is no space in this article to explain 6502 machine language.

Table 1. 6502 Addressing Modes

AD%()	Letter	Mode name	Length	Example
0	A	Immediate	2	LDA $2A
1	B	Zero page	2	LDA $2A
2	C	Relative	2	BNE $2A
3	D	Absolute	3	LDA $1E2F
4	E	Zero, X indexed	2	LDA $2A,X
5	F	Implied	1	DEX
6	G	Zero, Y indexed	2	LDA $2A,Y
7	H	Absolute, X	3	LDA $1E2F,X
8	I	Absolute, Y	3	LDA $1E2F,Y
9	J	Indexed Indirect	3	LDA ($1E2F,X)
10	K	Indirect Indexed	2	LDA ($2A),Y
11	L	Absolute Indirect	3	JMP ($1E2F)

CONTINUED OVER

The addressing modes are read from the DATA statements in lines 1000–1150. These contain the three-character mnemonics for each instruction, followed by a letter, A to L, which corresponds to the addressing mode. The opcodes are in numeric order, 16 to each BASIC line. Gaps in the sequence are compressed (rather than put ???F Hex for every invalid opcode). A number in the DATA represents the number of invalid opcodes in each gap. The SZ%() array contains the size (in bytes) of each type of instruction. Zero page addressing (MODE 1) requires two bytes per instruction and consequently,SZ%(1) = 2. Implied addressing (MODE 5) requires only one byte, so SZ%(5) = 1, and so on. Line 1160 contains the data which is loaded into the SZ%() array.

The fourth array is named SYMBOL$(). We gave the other arrays short names, in deference to the fact that you will have to type the program in! SYMBOL$-() contains the 31 label-names which are recognised by the disassembler. The list could be extended if the program were altered to recognise other labels. The addresses associated with each label are not stored. In principle, it would be simplest to check each disassembled address against a table of label locations, but that would require 31 or more comparisons before each address could be printed. The documented locations fall in two small blocks, so most addresses can be rejected after only four comparisons (ensuring that they are outside both blocks). Lines 570–580 carry out these tests.

There are very few quirks in the program. The VDU commands will only work properly in MODE 7 (teletext) graphics. FNHEX is used rather than the ROM hexadecimal output facility, since it was required that Hex numbers should be printed in neat columns, an even number of digits wide. In line 590 our printer has patriotically output a £ sign in place of the American hash symbol.

SECRETS REVEALED

The symbolic disassembler is a useful addition to the machine code programmer's library. It indicates the details of ROM routines, allowing existing code to be copied or used without the effort of designing and testing every section from first principles. You can even use the disassembler to find out the secrets of other programmer's creations!

PROGRAM LISTING

```
  100 REM ** BBC Micro disassembler **
  110 REM ** (c) 1982 Simon Goodwin **
  120 MODE 7
  130 VDU 130 : REM Print green message
  140 PRINT"Setting up. Please wait."
  150 DIM AD%(255),OP$(255),SZ%(11),SYMBOL$(30)
  160 PR%=0 : REM We're not printing
  170 NUM%=0
  190 REM ** Read in opcodes and modes
  200 FOR INSTRUCTION%=0 TO 255
  210 READ OPCODE$
  220 IF LEN(OPCODE$)=4 THEN OP$(INSTRUCTION%)=LEFT$(OPCODE$,3) : AD%(INSTRUCTIO
N%)=ASC(RIGHT$(OPCODE$,1))-65 : GOTO 270
  230 FOR DUMMY%=INSTRUCTION% TO INSTRUCTION%-1+VAL(OPCODE$)
  240 AD%(DUMMY%)=5 : OP$(DUMMY%)="???"
  250 NEXT DUMMY%
  260 INSTRUCTION%=INSTRUCTION%+VAL(OPCODE$)-1
  270 NEXT INSTRUCTION%
  280 FOR SIZE%=0 TO 11 : READ SZ%(SIZE%)
  290 NEXT SIZE%
  291 REM ** Build symbol table
  292 FOR SYM%=0 TO 30
  293 READ SYMBOL$(SYM%)
  294 NEXT SYM%
  300 CLS
  310 VDU 141 : PRINT TAB(7);"BBC Dissassembler 1"
  315 VDU 141 : PRINT TAB(7);"BBC Dissassembler 1" : VDU 140
  320 PRINT : VDU 131,136 :PRINT TAB(7)"By Simon Goodwin."'
  321 PRINT'"Press <SPACE> during disassemby to pause  the listing."
  322 PRINT'"Type <SPACE> again to continue or RETURN  to stop."
  323 PRINT'"Control characters such as control N and  control B work as normal.
"''
  330 INPUT"Start address",ADR$
  340 ON ERROR GOTO 300 : REM Trap eval
  350 P%=EVAL ADR$
  360 IF P%>65535 OR P%<0 THEN 300
  370 ON ERROR OFF
  380 INST%=?P%
  390 INSTSZ%=SZ%(AD%(INST%))
  400 IF P%+INSTSZ%>65536 THEN PRINT"END OF MEMORY";CHR$ 152 : GOTO 435
  410 GOSUB 500 : REM Disassemble
  420 PRINT LIN$
  430 IF INKEY$(0)<>" " THEN 450
  435 VDU 129 : PRINT"Press <SPACE> to continue"
  440 T$=INKEY$(0) : IF T$="" THEN 440 ELSE IF T$<>" " THEN 300
  445 VDU 11
  450 P%=P%+INSTSZ%
  460 GOTO 380
  490 REM Generate LINE$
  500 I%=0 : LIN$=FNHEX(P%) : IF LEN(LIN$)<3 THEN LIN$="00"+LIN$
  505 LIN$="&"+LIN$+"  "
  510 REPEAT LIN$=LIN$+FNHEX(?(P%+I%))+" ": I%=I%+1: UNTIL I%=INSTSZ%
  520 LIN$=LIN$+STRING$(3-INSTSZ%,"   ")+" "
  530 I%=0
  540 REPEAT J%=?(P%+I%)
  543 IF J%<32 OR J%>126 THEN J%=46 : REM Full stop
  546 LIN$=LIN$+CHR$(J%) : I%=I%+1: UNTIL I%=INSTSZ%
  550 LIN$=LIN$+STRING$(4-INSTSZ%," ")+OP$(INST%)+" "
  555 IF AD%(INST%)<>2 THEN 560 ELSE J%=?(P%+1) : IF J%>127 THEN J%=-256+J% : RE
M Relative jump
  556 J%=P%+J%+2 : IF J%<0 OR J%>65535 THEN DAT$="????" ELSE DAT$=FNHEX(J%) : IF
LEN(DAT$)<3 THEN DAT$="00"+DAT$ : REM Zero-page
  557 GOTO 572 : REM Check symbols
  560 REM
  561 IF INSTSZ%=1 THEN 580 ELSE DAT$=FNHEX(?(P%+1))
  570 IF INSTSZ%>2 THEN DAT$=FNHEX(?(P%+2))+DAT$
  571 IF INSTSZ%=1 THEN 580
  572 DAT%=EVAL("&"+DAT$)
  573 IF DAT%<&200 OR DAT%>&223 THEN 576 : REM Not a vector
  574 DAT$=SYMBOL$((DAT%-&200) DIV 2) : IF DAT% MOD 2=1 THEN DAT$=DAT$+"+1"
  575 GOTO 580
  576 IF DAT%>&FFCD AND DAT%<&FFE4 AND DAT% MOD 3=2 THEN DAT$=SYMBOL$(((DAT%-&FF
CC) DIV 3)+18) : GOTO 580
  577 IF DAT%=&FFE7 THEN DAT$=SYMBOL$(26) : GOTO 580
  578 IF DAT%>&FFED AND DAT%<&FFF8 AND DAT% MOD 3=1 THEN DAT$=SYMBOL$(((DAT%-&FF
EE) DIV 3)+27)
  580 ON AD%(INST%)+1 GOTO 590,600,610,610,620,630,640,650,660,670,680,690
  590 LIN$=LIN$+"£ &"+DAT$ : RETURN
  600LIN$=LIN$+"$ &"+DAT$ : RETURN
  610 LIN$=LIN$+"&"+DAT$ : RETURN
  620 LIN$=LIN$+"$ &"+DAT$+",X" : RETURN
  630 RETURN
  640 LIN$=LIN$+"$ &"+DAT$+",Y" : RETURN
  650 LIN$=LIN$+"&"+DAT$+",X" : RETURN
  660 LIN$=LIN$+"&"+DAT$+",Y" : RETURN
  670 LIN$=LIN$+"($ &"+DAT$+",X)" : RETURN
  680 LIN$=LIN$+"($ &"+DAT$+"),Y" : RETURN
  690 LIN$=LIN$+"( &"+DAT$+")" : RETURN
  700 PRINT"WHOOPS!"
 1000 DATA BRKF,ORAJ,3,ORAB,ASLB,1,PHPF,ORAA,ASLF,2,ORAD,ASLD,1
 1010 DATA BPLC,ORAK,3,ORAE,ASLE,1,CLCF,ORAI,3,ORAH,ASLH,1
 1020 DATA JSRD,ANDJ,2,BITB,ANDB,ROLB,1,PLPF,ANDA,ROLF,1,BITD,ANDD,ROLD,1
 1030 DATA BMIC,ANDK,3,ANDE,ROLE,1,SECF,ANDI,3,ANDH,ROLH,1
 1040 DATA RTIF,EORJ,3,EORB,LSRB,1,PHAF,EORA,LSRF,1,JMPD,EORD,LSRD,1
 1050 DATA BVCC,EORK,3,EORE,LSRE,1,CLIF,EORI,3,EORH,LSRH,1
 1060 DATA RTSF,ADCJ,3,ADCB,RORB,1,PLAF,ADCA,RORF,1,JMPL,ADCD,RORD,1
 1070 DATA BVSC,ADCK,3,ADCE,RORE,1,SEIF,ADCI,3,ADCH,RORH,1
 1080 DATA 1,STAJ,2,STYB,STAB,STXB,1,DEYF,1,TXAF,1,STYD,STAD,STXD,1
 1090 DATA BCCC,STAK,2,STYE,STAE,STXE,1,TYAF,STAI,TXSF,2,STAH,2
 1100 DATA LDYA,LDAJ,LDXA,1,LDYB,LDAB,LDXB,1,TAYF,LDAA,TAXF,1,LDYD,LDAD,LDXD,1
 1110 DATA BCSC,LDAK,2,LDYH,LDAE,LDXI,1,CLVF,LDAI,TSXF,1,LDYH,LDAH,LDXI,1
 1120 DATA CPYA,CMPJ,2,CPYB,CMPB,DECB,1,INYF,CMPA,DEXF,1,CPYF,CMPD,DECD,1
 1130 DATA BNEC,CMPK,3,CMPE,DECE,1,CLDF,CMPI,3,CMPH,DECH,1
 1140 DATA CPXA,SBCJ,2,CPXB,SBCB,INCB,1,INXF,SBCA,NOPF,1,CPXF,SBCD,INCD,1
 1150 DATA BEQC,SBCK,3,SBCE,INCE,1,SEDF,SBCI,3,SBCH,INCH,1
 1160 DATA 2,2,2,3,2,1,2,3,3,3,2,3
 1170 DATA USERV,BRKV,IRQ1V,IRQ2V,CLIV,BYTEV,WORDV,WRCHV,RDCHV,FILEV,ARGSV,BGETV
,BPUTV,GPBV,FINDV,FSCV,EVNTV,UPTV
 1180 DATA OSFIND,OSGBPB,OSBPUT,OSBGET,OSARGS,OSFILE,OSRDCH,OSASCI,OSNEWL,OSWRCH
,OSWORD,OSBYTE,OSCLI
 2000 DEF FNHEX(NUM%) LOCAL T$,N2%
 2010 REPEAT
 2020 N2%=NUM% MOD 16
 2030 NUM%=NUM% DIV 16
 2040 IF N2%<10 THEN T$=CHR$(N2%+48)+T$ ELSE T$=CHR$(N2%+55)+T$
 2050 UNTIL LEN(T$) MOD 2=0 AND NUM%=0
 2060 =T$
```

Our Software Competition

Win £100's worth of software for your BBC Micro in this easy-to-enter competition.

To celebrate our first issue of *A & B Computing*, we want to give away £100's worth of software to one of our readers. But not to just anyone—you'll have to run the gauntlet of the word-square below first.

No doubt you'll have seen a wordsquare before, and this one's no different to the rest! Hidden in among the maze of jumbled-up letters, you'll find 16 BASIC keywords indiginous to the BBC Micro. The words you have to find have been written diagonally, vertically and horizontally, and any other way we could think of (without cheating of course!) to make them especially difficult for you to find.

Once you think you've found all 16 keywords, count up the number of shared letters and write your answer on the back of the envelope you are going to send your entry in on. Then, once you have clearly indicated the keywords from the rest of the letters, post if off to the following address:

A & B Computing Competition,
145 Charing Cross Road,
London WC2H 0EE.

SOFT OPTIONS

The winner of the competition will be the first correctly completed wordsquare picked at random and will receive his or her choice of up to £100's worth of software from the latest range offered from software manufacturer Micro Power.

The winner will be notified once the competition has closed and the results will be published in a future issue of *A & B Computing*.

RULES

This competition is open to all UK and Northern Ireland readers of *A & B Computing*, except employees of Argus Specialist Publications Ltd, their printers and distributors, employees of Micro Power Ltd, or anyone else connected with the competition.

As long as the correct coupon is used for each entry, there is no limit to the number of entries per person.

All entries must have the number of shared letters written on the outer flap of the envelope. Entries without this number will not be accepted.

All entries must be postmarked before June 30, 1983.

The prize will be awarded to the first correct entry picked at random which satisfies the rules of the competition.

No correspondence will be entered into with regard to the results and it is a condition of entry that the Editor's decision is accepted as final.

The winner will be notified by post and the results will be published in a future issue of *A & B Computing*.

Sorry, the bottom half of this page was cut out

David from the Retro Computer Museum, Leicester

www.retrocomputermuseum.co.uk

Software for the BBC Micro

....ROMS....

Available Soon! SEND FOR DETAILS

Beeb-calc

A ROM based spread sheet program, like Wordwise this firmware is very fast and simple to use — yet is a powerful spread sheet analysis program, considerably better than the original 'calc' program — full floating point maths. Works in 80 or 40 column screen modes, variable column widths. Works with either cassette or disk.
This ROM coupled with Wordwise can turn your micro into an ideal small business machine.
£34.00 + VAT

Debugging Program

2 machine code programs — one in ROM, one on tape. Essential for the machine code programmer. An ideal compliment to the assembler built into the BBC machine. Contains a full machine code monitor allowing examination and alteration of memory, registers, setting of break points and even single stepping through machine code programs.

ROM based £19.00 + VAT
cassette £10.00 + VAT

Disk Doctor

A ROM containing useful disk utility programs. Enables the recovery of any data off the disk including deleted files etc. The full disk editor allows the alteration of any bytes directly on the disk (or in memory), or the loading and saving of any track or sector on the disk. Automatic transfer of programs from tape to disk and visa versa.
Includes a whole host of other useful utilities — string search, function key editing, the ability to format 35, 40 & 80 track disks.
£19.00 + VAT

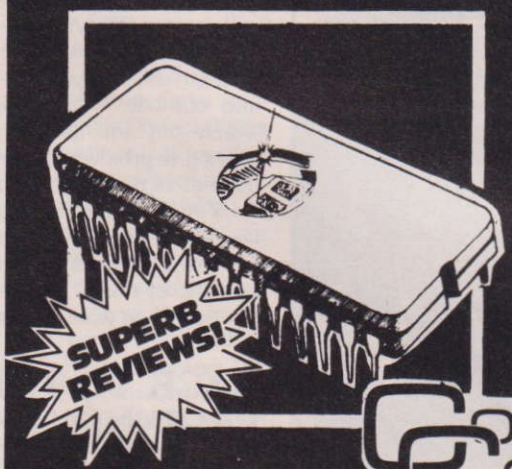
Printer toolkit ROM

This includes routines for high resolution screen dumps for both the EPSON and NEC printers. Will work in any graphics mode with automatic grey shading of all screen colours. The most useful feature of this program is its 'spooling' capability, this enables data such as a program listings or high res screen dumps to be automatically spooled from your disk to the printer **while** using your BBC machine for running other programs.
£19.00 + VAT

Quantity discounts, starting at five off.

Wordwise

THE WORD PROCESSOR FOR THE BBC MICRO



Wordwise

The word processor for the BBC machine.
This ROM based word processor has received superb reviews.
A powerful and flexible system — it's greatest strength being that it is fast, simple and very easy to use. No other existing word processor (even ROM based, costing twice as much) can compare with the flexibility of this system.
Supplied with full spiral bound manual and cassette containing an example document and free typing tutor program.

£39.00 + £1.50 p&p + VAT

Now available from stock. Quantity discounts, starting at five off.

all ROM based software requires new series 1 Operating Systems. We are now in the position to supply 1.2 ROM's for £5.00 + VAT when you buy any of the above software.

CASH OR ROYALTIES waiting for any interesting software (not just games) for the BBC Micro. The better the program the more we will pay.

Computer Concepts

Dept **BC1**
16, Wayside, Chipperfield,
Herts, WD4 9JJ. tel (09277) 69727

Buy it with Access

BARCLAYCARD VISA

Cells and Serpents

IG Nichols

Dare you discover the dark secrets of the dungeons in this epic adventure game?

Cells and Serpents is one of the classic implementations of adventure games in the 'Dungeons and Dragons' genre. Now converted for the BBC Micro, here is a new adaptation of this listing for you to delve into the depths of the dungeons.

You start the game searching through a randomly generated sequence of rooms, closed doors and corridors all on level 1. Your aim if to find the exit which is somewhere amongst the infinite maze of passages that lead on down through the various levels; don't feel despondent if you don't find the exit immediately—your chances of finding it at any point in the maze are one in 237!

This version of the game has been written to conserve storage space and consequently only allows you to move forwards as you explore the unknown—you may not retrace your steps.

STEPPIN' OUT

At each move, you are given three choices of going left, forwards or right; to move in one of these directions you should press 'L', 'F' or 'R' keys to indicate your choice. For each direction indicated, you are told that should you move you will either enter a corridor, a door, a blank wall, a room, stairs leading up, stairs leading down, or you have found the exit.

If you have chosen a corridor, you will be presented with three more directions to move in. Should you choose to move to a door, you will be invited to open it, or to be cautious and listen (of course, this doesn't mean you'll be told definitely whether something is laying in wait for you!). If you open the door it may well be empty, but it may also contain treasure or a monster, or both.

Should you come across a monster, you can stay and fight, cast a spell on it or retreat; the monster will, however, sometimes attack before you can use the latter option. You are able to find out the monster's strength (measured in hit-points) and your own status (comprising hit-points, spells left, number of gold pieces, monsters killed, the level you are currently exploring, your score and whether or not you have a luckstone). If your hit-points are greater than the monster's, you will be able to beat it but your score will be reduced by the amount of the monster's hit-points. If you choose to cast a spell on the monster to subdue it, you will only reduce the number of spells you have left and your hit-point level remains untouched.

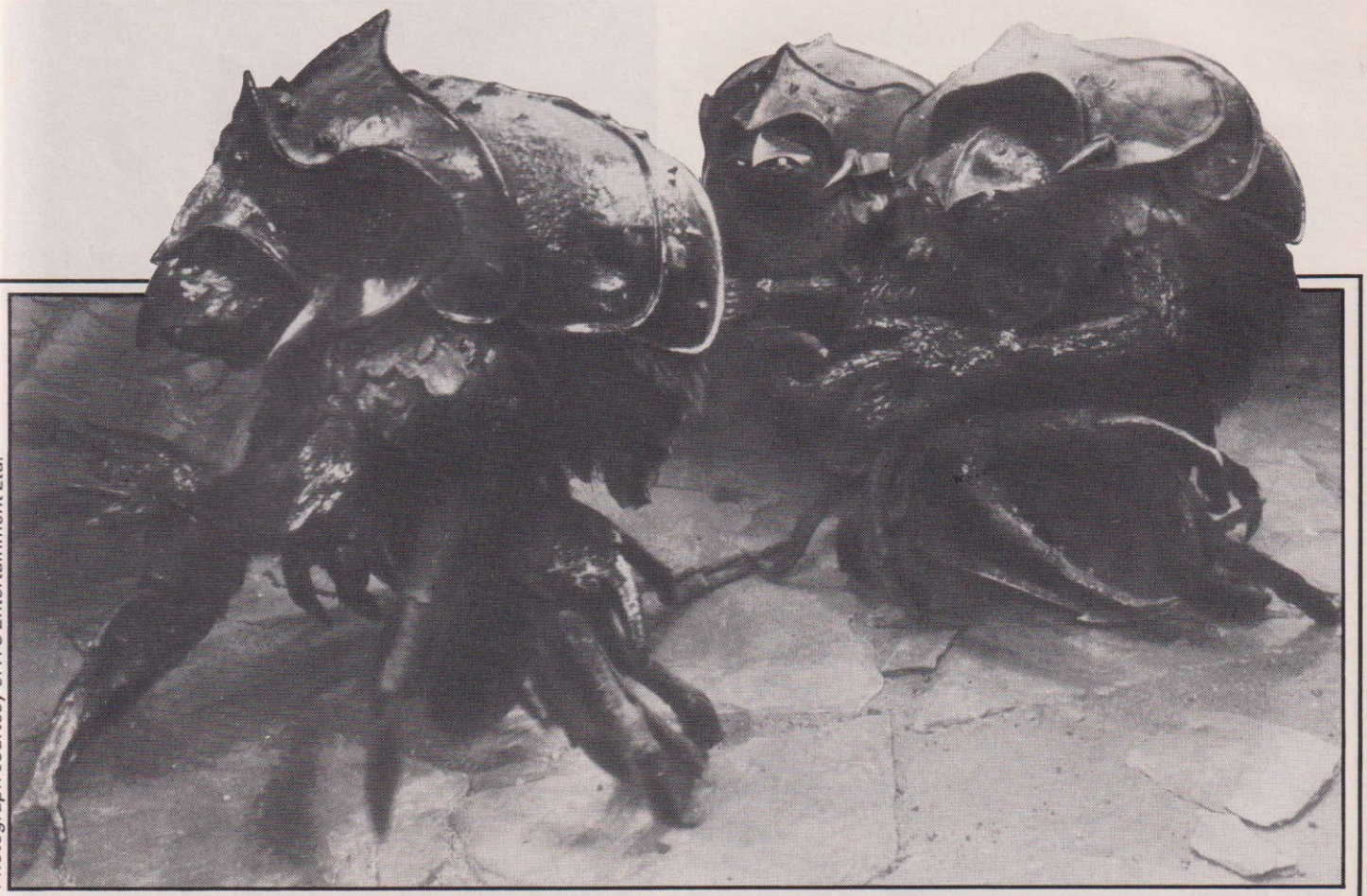
Photograph courtesy of Dino De Laurentis Corporation.

When you come across stairs, you have the opportunity to move between levels of the labyrinth. The lower the level you are on when you kill a monster, the greater will be the resulting increase in your score. However, the monsters are a lot tougher the deeper you delve. Also, from time to time, the monsters will appear unexpectedly and you will not always have the opportunity to investigate their strength before you are thrust into combat. You should also watch out for hidden pits of varying depths which contain all manner of unspeakable horrors.

You shouldn't need any warning from me that walking into blank walls is not the most sensible of things to try—if you do, you'll see what I mean!

BACK TO THE BEGINNING

The program started life as a version written for the NASCOM microcomputer (in pre-Lucas days) to be run using the CCSOFT BASIC interpreter. Cells and Serpents has been extensively overhauled for this article in order to produce a version which makes good use of



Photograph courtesy of ITC Entertainment Ltd.

the advanced features of BBC BASIC. From the user's point of view, the program listed in this article is identical to the version currently available from ASP Software. However, if you LIST that version I think you'll agree that this particular adaptation is a lot easier to follow.

The main reason it is easier to follow is that extensive use has been made of procedures with descriptive names (in fact, there are no subroutines—as used in the original program—left at all). A lot of detail is presented in the rest of the article so that you can see just how the program works. This will be of great value to you if you want to write similar adventure programs yourself.

There is one remaining link with Cells and Serpents' origins and that is the array A(80). In CCSOFT NASCOM BASIC you can only have one single dimension array called '@'. There did not seem much point in splitting this down into a number of other arrays so it is still there, except that its name has been changed to A!

One last point before we start delving into the programs, and that is the way it has been LISTed. First, it has been LISTed using the 'LIST07' command. This inserts a space between the line number and the rest of the line and it also indents REPEAT . . . UNTIL and FOR . . . NEXT loops.

Second, the main program lines increase in steps of 10, and REM statements have been inserted with intermediate line numbers. This has been done to allow those of you who wish to copy it to do so easily using AUTO (to generate line numbers automatically) and to be able to avoid typing in the comments. The program will run on a Model A, providing the REM statements are left out.

PROGRAM DESCRIPTION

Lines Description

10-20 The BBC Micro makes very full use of the screen area on a monitor or domestic TV. This is in contrast to machines like the VIC 20, the ZX Spectrum or the Atari which have a substantial coloured border around the active part of the screen. On a monitor, this aspect of the BBC Micro causes no problems, but most domestic TVs are set up with a substantial overscan; that is, the edges of the transmitted picture are lost around the edge of the screen. One reason for doing this is that whereas the transmitted picture is a rectangle, the edges of a TV picture tube are not. In order to see the whole of the transmitted picture there would be blank areas around the edges. The other reason is that at the top of the picture there are two lines carrying the Ceefax or Oracle Teletext information and the ever-changing patterns on these lines would be distracting if they were visible. The net result of this overscan on most TVs is that either the top of bottom of the picture produced by the BBC Micro cannot be seen; usually it will be the top. There is a Machine Operating System (MOS) command which allows us to move the picture up or down the screen—the command is '*TVa,b'.

The second parameter of the command, b, controls whether the picture is interlaced or not, but this need not concern us for the moment. If the first parameter is 1, the picture will be moved up by one line, if it is 2 it will be moved up by two lines, and so on. Consequently, if its value is 255, the picture will be moved down by 1 line, if it is 254 it will be moved down by two lines, etc. In this context, a 'line' is one row of the screen display which will have either 25 or 32 rows, depending on the mode we are in. The result of the '*TV' command does not take effect until the next mode change, so line 10 asks for

CONTINUED OVER

Photograph courtesy of Henson Associates, Inc.

the screen display to be moved down one line and line 20 actually brings it into effect. (The second parameter ,b, is zero and can, therefore, be left off.)

30 This is a DATA statement containing 14 values which is used by the procedure, PROCmonster. We will look at what it does when we come to that procedure.

40 **PROCtitle:** This statement calls up the procedure named PROCtitle which is at the end of the program from line 3720 to line 3840:

3730 Dimensions array (A(80).

3740 Initialises the various 'flags' used in the program.

3750 Seeds the random number generator with a value based on the value of the internal clock, TIME. Since TIME is very unlikely to have the same value at this point in the program every time you RUN it, this line ensures that each RUN of the program will use a different series of random numbers.

3760 Turns the flashing cursor off before the title page is drawn.

3770-3780 Place the ASCII control codes 135 and 157 at the beginning of each line on the screen. The effect of this is to produce a white background (135-code for white alphanumeric characters, 157-code for a new background).

3790-3800 Print the title 'CELLS AND SERPENTS' in double height (Control code 141) and in blue letters (Control code 132).

3810 Calls the procedure PRO-delay (see later).

3820 Restores the flashing cursor, clears the screen and returns from the procedure.

50 **PROCload:** The procedure PROCload is now called; this is also at the end of the program, from line 3850 to line 3990:

3800-3870 Ask if you wish to load a character from tape.

3880 Waits for a key to be pressed, and if the key is not 'Y' or 'N' reprints the last line of the question.

3890 Calls procedure PRO-read_character if 'Y' was pressed (see later). If 'N' was pressed then no action is taken.

3900 Returns from the procedure.

This is the end of the initialisation phase, and we can now move on to the program proper.

THE MAIN PROGRAM LOOP

The program works by cycling repeatedly through a loop. The purpose of the loop is to generate the next set of three choices to move into, and then to follow through with the consequences of the choice that you make. When all of these consequences have taken place, it generates the next set of three choices, and so on. The process finishes when one of three things happens:

- you run out of 'hit-points', the measure the strength used when you fight a monster (in other words, you are dead!). You are then offered a choice of starting all over again.
- you decide to end the game yourself by SAVEing your character on tape.
- you find the exit. At this point you can decide to finish the game, or return to the first level again, but with your current score, hit-points, stock of spells, treasures and luck-stone (if you have one), left intact.

The main program loop runs from line 60 to line 140. It is a REPEAT . . . UNTIL loop, with the condition in the UNTIL statement being 'until TRUE=FALSE'. Since this condition will never be met, the loop will continue for ever. However, as mentioned above, statements in the body of the loop will at some stage either bring the game to an end, or cause it to start all over again. For those of you who find flow charts easier to follow than program listings, Fig. 1 is a flow chart of the main program loop. Let's take the loop line by line.

Line 70 - Procbuild 1: CLS

PROC build 1_(lines 390-540): The first three elements of array A(80), that is A(1) to A(3), are used to indicate what lies in the directions 'LEFT', 'FORWARD' and 'RIGHT'. Each of them contains a value from one to seven—the meanings of these values are:

1 Corridor.
2 Door.
3 Blank wall.
4 Room.
5 Stairs up.
6 Stairs down.
7 Exit.

Procedure PROCbuild_1 chooses the values to be placed in A(1) to A(3).

For each of the three array elements, A(1) to A(3), a random value is chosen, A1, which is then subjected to a number of tests:

420 If A1 = 1 to 4, A1 is stored in the array element.

430 If you are on level 1 (L1=1) and both the 'LEFT' and 'FORWARDS' directions are either a blank wall or stairs up, then 'RIGHT' (ie A(3)) is made a corridor.

440 If A1 = 5 (stairs up), then this is only used nine times in 100 (RND(100) < 10).

450 If A1 = 6 (stairs down), then this is only used 24 times in 100 (RND(100) < 25).

460 If A1 = 7 (exit), then this is only used one time in 100 (RND(100)=1).

470 If any of the tests in lines 430 to 460 have failed, this line chooses a random value between one and four for A1 and stores it in the array element.

If any of the elements A(1) to A(3) contain two or four, then we need to determine what lies behind the door or in the room. Lines 500 to 530 do this test and procedure PROCbuild_2 is called to determine what the room contents are.

PROCbuild_2 (lines 550-630): Elements A(4) to A(6) of the array A(80) are used to hold the room contents of A(1) to A(3).

560 Puts random numbers between one and four into A(4) to A(6).

570-580 Test to see if values of one and four were chosen, if so the procedure is ended.

590 If values of two or three were chosen, then the room will hold a monster, which is selected by PROCmonster.

PROCmonster (lines 2580 to 2630): This procedure reads the 14 numbers in the DATA statement (line 30) and generates 14 random numbers based on them. These numbers are stored in A(10) to A(23).

600-620 Copy the contents of A(10)-A(23) into A(30)-A(43), A(45)-A(58) or A(60)-A(73) depending on whether the room or door is on the 'LEFT'(D1=1), 'FORWARDS'(D1=2) or on the 'RIGHT'(D1=3).

Line 80—PROCmonster

As we have seen above, this procedure puts 14 random numbers into A(10) to A(23). If PROCbuild_2 has already used PROCmonster, then the values already in A(10) to A(23) will be overwritten.

Line 90—IF A(1)=3 AND A(2)=3 AND A(3)=3 PROCpit: GOTO 140

If PROCbuild_1 has chosen a blank wall for each of the three directions 'LEFT', 'FORWARDS' and 'RIGHT', then PROCpit is called, since otherwise there would be no way to move. With this procedure you fall down a pit thereby losing hit-points.

PROCpit (lines 150-280)

160-180 Calculate the depth of the pit from 10 to 120 feet, the amount of damage (in hit-points) that you incur and print out the appropriate messages.

190 Inflicts further damage on you if RND(6)=1, by jumping to line 210.

200 Decrements your hit-points (H1=H1-D1), takes you down one more level (L1=L1+1), waits for a few seconds (PROCdelay(20)), prints out your status with PROCstatus 1 and then exits the procedure.

210-220 Tell you that worse is to come and choose which of three possibilities it will be.

230-240 Tell you what hit-points you have lost from landing in a pool of acid or on some spikes, then jump back to line 200 to exit the procedure.

250 Chooses a monster, prints out its name and increases the hit-points you will lose if you have been unlucky enough to be attacked by one of the 'super monsters'. You then have to choose whether to fight the monster with PROCcombat, before exiting the procedure. Lastly, GOTO 140 jumps to the end of the main program loop and we start again with line 70.

Line 100—PROCoptions: PROCoperate_cell(1)

To have reached this line we have avoided the pit and we now have the movement choices displayed on the screen by PROCoptions (lines 740-860). The headings 'LEFT', 'FORWARDS' and 'RIGHT' are printed out in white, then underneath each one is printed what lies in that direction. 'Corridor', 'Door', 'Room', 'Stairs up', and 'Stairs down' are all printed in green. 'Blank wall' is printed in red, and 'Exit' is printed in flashing yellow. The colours are produced using the following Teletext Control characters in the PRINT statements:

CHR$130 — green
CHR$131 — yellow
CHR$129 — red
CHR$136 — flashing

Since all these messages are printed on the same line, and a Teletext Control code applies to the end of the line (or until another code changes its effect), line 840 prints CHR$137 after printing 'Exit'. This turns off the flashing effect.

The next procedure, PROCoperate_cell (I%), is the heart of the program and extends from line 870 to line 1660. The main program loop calls it twice, once in the present line (100) and once in line 120. However, different entry points to it are used by means of parameter I%. Line 120 calls PROCoperate_cell(2) after a monster has suddenly arrived from nowhere. Let's now look at the procedure PROCoperate_cell(I%) (lines 870-1660) in a bit more detail:

880 Jumps to line 890 or 1550, depending on the value of I%.

890 Asks you to press keys 'L','F' or 'R' to indicate the direction you have chosen, or 'S' to see your current status (see PROCstatus_1). It also sets a counter, SS, to 0. This counter is increased by one each time you cast a spell (see line 1740). When SS=2, the screen is cleared to avoid the text scrolling (see line 1500).

900 Assigns values to the four possible choices.

910 Calls PROCread_keyboard to read a key (see below) and then puts the ASCII code of the key pressed into D1.

920 The ASCII codes for 'L', 'F','R' and 'S' are 76, 70, 82

CONTINUED OVER

and 83 respectively, so this makes D1 equal to 1,2,3 or 4 according to which key has been pressed (or zero is any other was pressed).

930 Calls PROCread_1 if 'S' was pressed and then jumps back to line 890.

940-950 Jump back to line 890 if a different key from 'L', 'F','R' or 'S' was pressed.

960-990 Test to see if there is a monster waiting in the direction chosen (placed there by PROCbuild_2). If there is then the random numbers which specify which monster it is are loaded back into A(10)-A(23).

1000 Tests to see if we have found the exit, and jumps to PROCexit(I%) if we have. If not it jumps to one of six lines depending on what lies in the direction we have chosen.

Line 1010—Corridor.

The procedure is ended straight away and we move to line 110 of the main program loop.

Line 1020—Door.

1020-1080 Offer you three choices of 'Open', 'Listen' and 'Status' which are handled as in lines 850-950 described above.

1090 Jumps to the beginning of the routine which handles the 'Room' option if you choose to open the door.

1100-1120 Print out 'You hear' and, if there is no monster in the room (R1<>2 and R1<>3), print out 'nothing', then jump to line 1160.

1130-1150 On average will also print out 'nothing', two times in every three even if there *is* a monster there! On the one time in three that it does tell the truth the program prints the monster's name (PROC-print_monster) and then tells you what sort of noise it's making (PROC-monster_noises).

1160-1220 Again present you with three choices of opening the door—'yes', 'no' or 'status'.

1230 Jumps to the routine which handles the 'Room' option if you choose 'yes'.

1240 Tells you you are 'Chicken' if you choose not to open the door, re-presents the three movement choices, and jumps back to line 890.

Line 1250—Blank Wall

1250 Prints a suitably unsubtle message in flashing yellow characters.

1260 Lets you get away with it one time in six, displaying the movement options again and jumping to line 890.

1270-1310 However, five times out of six, the wall falls on

you and you lose up to 20 hit-points.

1320 Your status is then displayed.

Line 1330—Room

1330 Sets R1 equal to the room contents created by PROCbuild_2.

1340 Prints 'The room'.

1350 If R=1, then the room is empty and the procedure is exited.

1360-1370 Print 'contains' and tests to see if R1=2. If it does then the room contains a monster, whose name is then printed.

1380 If R1=5, the room contains treasure and a monster.

1390 If R1=4, the room contains only treasure.

1400-1460 Offer four choices of 'go in', 'leave', 'help' and 'status'. 'Help' will tell you the hit-points of the monster. 'Status' will tell you your remaining hit-points, ie whether you can beat the monster in a fight. These lines read your choice as in lines 850-950 (qv).

1470 If you decide to leave the program, it jumps to the line used by the door routine in similar circumstances (line 1240).

1480 If you've asked for help and the room does contain a monster, you are told the monster's hit-points, and you are then asked to make you choice of movement again (jump to line 1400).

again (jump to line 1400).

1490 If R1=4, the PROCtreasure(I%) is called with I%=1 (see later) and the procedure is exited.

1500 'go in' is the only option left; the print loop counter, SS, is set to zero if it equals two, and there is a short delay.

1510-1540 You are now asked to make a further choice between 'combat', 'spell cast' and 'retreat'; the choices are handled as in lines 850-950 (qv).

1550 If you chose to 'retreat' then you jump to line 1590 (qv).

1560 If you choose 'combat', PROCcombat is called (qv), and the procedure is exited.

1570 If you choose to 'cast a spell', this line checks to see if you have any left; if not, you are asked to make a new choice (jump to line 1500). If you have spells left, PROCcast_spell is called (qv) and if the monster hasn't attacked you (F%=1), you have to make another choice on what to do (jump to line 1500); otherwise the procedure is exited.

1580 Jumps back to line 1500, because you did not choose any of the three valid choices.

1590 You have jumped here because you chose to 'retreat'; one time in six, you will be attacked anyway and the procedure exited.

1600 Tests to see if you have got here because of the sudden arrival of a monster rather than finding one in a room (RM=1); if so, a suitable message is printed out before exiting the procedure.

1610 If RM=0, jumps back to line 1240 and another suitable message.

Line 1620—Stairs up

1620 Reduces the level by one.

1630 Ends the procedure if the level is not zero.

1640 The level is zero, and you are told that the stairs are blocked, the level is increased to one again and there is a delay.

1650 The screen is cleared and you are returned to the beginning of the procedure to make a new choice (line 890).

Line 1660—Stairs down.

The level is increased by one, and the procedure is then exited.

Line 110—IF RND(12)<>1...

If RND(12)<>1, ie 11 times out of 12 on average, the program now jumps to line 130. One time out of 12, a monster suddenly arrives before we are given the next three choices. In this case, the monster is chosen (PROCmonster) and its name is loaded into B$ by PROCprint_monster.

Line 120—PRINT AS . . .

This line makes the 'room contents' flag, R1, equal to two and the 'sudden arrival of a monster' flag, RM, equal to one. It then jumps into PROCoperate cell(I%) at the second entry point with I%=2. This entry point is line 1500 (qv).

Line 130—V1=0:RM=0

The temporary storage variable for hit-points, V1, used in PROCcast spell and PROCcombat, is set to zero. The sudden arrival flag, RM, is also set to zero.

Line 140—UNTIL FALSE

The condition 'TRUE=FALSE?' will never be met so the program now jumps back to the start of the REPEAT . . . . . UNTIL loop (line 60).

TO PROCEED . . .

There are still some procedures that we have not examined, so let's have a look at them now.

PROCdelay(I%)—lines 290 to 320: This introduces a variable delay by cycling through a FOR . . . . .NEXT loop.

PROCstatus_1—lines 640 to 730: Your hit-points, remaining spells, the level you are currently on, your amassed teasure, the number of monsters you have killed and your score are printed out (in blue on a white background). If you have found a luckstone, then its value is also printed out. It then calls PROCstatus_2.

PROCstatus_2—lines 1900 to 2020: If your hit-points are negative then your're dead and line 1910 calls PROCdead. Line 1920 checks to see if you have found the exit (exit flag, E, equals one); if not, then it jumps to line 1980 which asks whether you want to continue or SAVE the character on tape. If you have found the exit, your hit-points and spells are increased if they are below set minimum values (lines 1930 and 1940). You are placed back at level 1, the exit flag (E) is set to zero and suitable messages are printed.

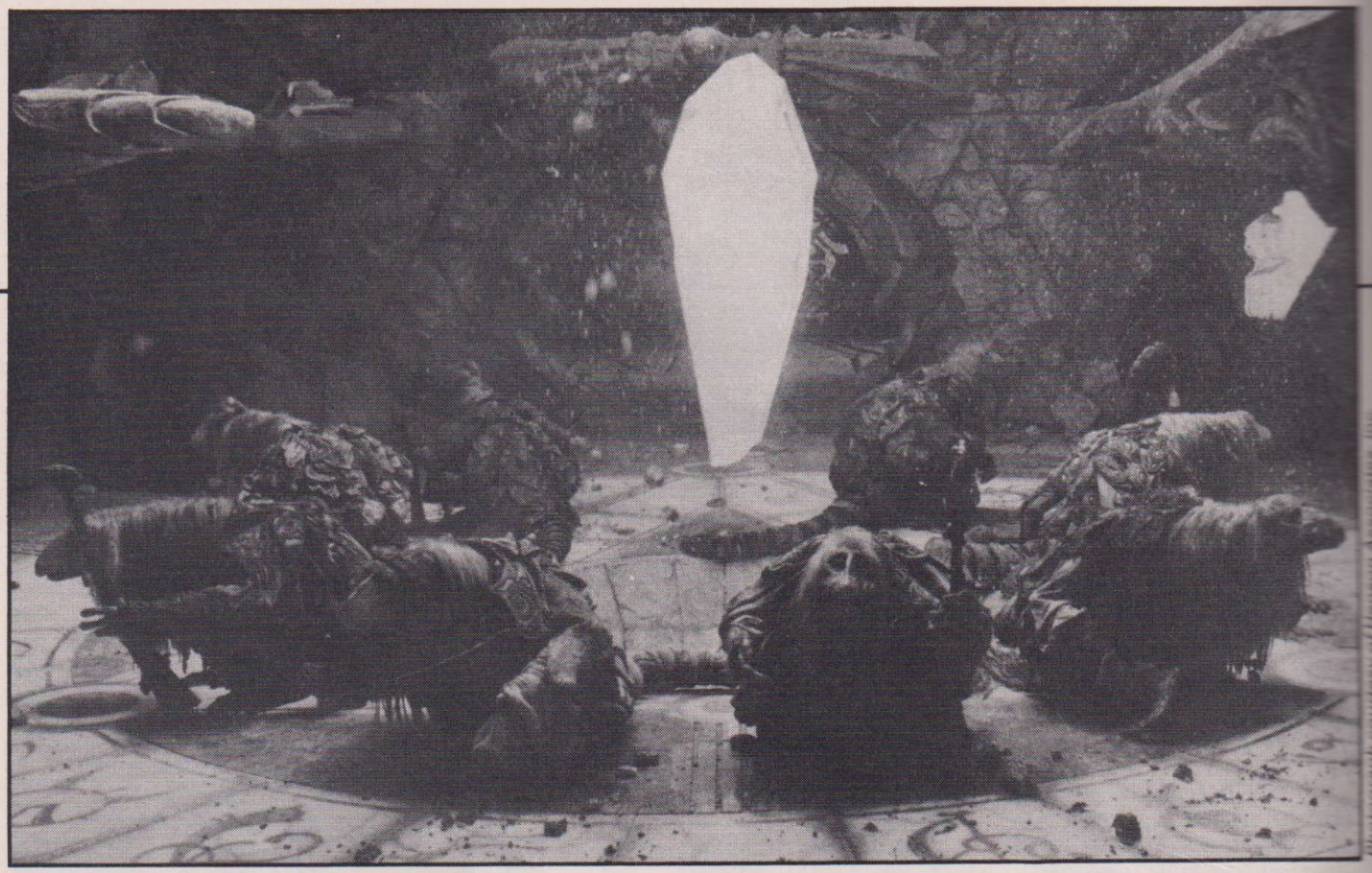
PROCdead—lines 2230 to 2270: You are here given the choice of playing again or finishing.

PROCcast_spell—lines 1720 to 1830: F% is first of all set to zero; this flag indicates whether the monster has attacked you (F%=5 if it has not). Your remaining spells are reduced by one and the monster's name is included in an appropriate message. A random number between one and 21 is chosen and the monster's hit-points are reduced by this amount. If its hit-points are less than one, you are told it is dead, the number of monsters killed is incremented (M1) and your score (SC) is increased by the value of the level you are on. Before exiting, PROCtreasure(2) is called.

If the monster's hit-points are greater than one, the monster advances on you. One time in three (RND(6) 3), the monster then attacks, and PROCcombat is called before exiting the procedure. For the remaining two times in three, the monster's hit-points are put in a temporary storage variable (V1) and F% is set to one. The procedure is then ended.

PROCcombat—lines 1840 to 1890: You are first told that you have had a 'terrific battle' with

CONTINUED OVER



the monster. Then, if the monster's hit-points are greater, you are told you have won; your tally of monsters killed, your hit-points and your score are then all updated and PROCtreasure(2) is called. Otherwise, you have been killed, you lose all your gold pieces (G1=0), your hit-points are updated and PROCexit(2) is called.

PROCtreasure(I%)—lines 2280 to 2540: If I%=1 and T1=0, then the treasure was an illusion! Line 1490 of PROCoperate_cell follows this path. If I%=2, then this test is bypassed. Line 1880 of PROCcombat sets I% equal to two. If R1=2 (ie the room contains a monster, but no treasure) the procedure finishes.

The number of gold pieces is then calculated based on the 'treasure points' of the monster, T1, you have just killed (T1=500 if you haven't just killed one—line 1390 of PROCoperate cell), on the level, L1, a random number between zero and one and on the value of your luckstone, J1, if you have one. Your score is increased by the number of gold pieces divided by 100.

The treasure flag, T, is now tested. If it is zero and RND(100)< (15+2* the level you are on) then you are given one of nine possible further treasures. Otherwise, if T=0, that is the end of the procedure. If T>0 then you will be given 'T' treasures. Some monsters, the nastier ones, set 'T' to values as high as seven—an 'Orcus', for example.

There is one chance in 10 of obtaining a luckstone, and a similar chance of getting an extra treasure.

PROCprint_monster—lines 2640 to 3010: PROCmonster puts 14 random numbers into A(10) to A(23). This procedure uses the values in A(10) and A(11) to choose one of 35 different monsters. For most of the monsters, it also sets the values of their hit-points, D1, their 'treasure-points', T1, and their 'treasure flag', T. For the other monsters, further procedures are called to choose one of a subset of monsters. These subsets are as follows:

PROCserpents—This uses A(12).
PROCdemons—This uses A(13), A(14) and A(15).
PROCdevils—This uses A(16), A(17) and A(18).
PROCgiants—This uses A(19).
PROCelementals—This uses A(20).
PROCgolems—This uses A(21).
PROClich—This uses A(22).
PROClycanthropes—This uses A(23).

All these monster procedures put the name in AS and one 'a', 'an' or ' ' into BS=" " for monsters with proper names, such as 'Asmodeus'.

PROCexit(I%)—lines 1670 to 1710: If I%=1, then you have found the exit, if I%=2, you have found the alternative exit—you are dead (line 1890 of PROCcombat)! The exit flag, E, is set to equal to one. This is checked in line 1920 of PROCstatus_2.

PROCread_character—lines 2130 to 2220
PROCsave character—lines 2030 to 2120

These procedures read from, and save to tape, your character. The values stores are your name plus:

H1—Your hit points.
S1—The number of spells left.
L1—The level you are on.
G1—the number of gold pieces you have.
M1—The number of monsters you have killed.
J1—the value of any luckstone you have.
SC—Your score.

PROCread_keyboard: This procedure clears the keyboard buffer (*FX15,1) and then reads the next key depression, putting the ASCII code into the variable 'key'. The corresponding character is then printed.

HAPPY ADVENTURING!

Well that is the end of the program! I hope you have been able to follow how the program works, and that you are just itching to type it in to your computer. There is obviously scope for development of the program; for example, there could be a greater use of different background colours, double height letters and imaginative Teletext graphics. Those of you with 32K of memory could attempt to draw pictures to illustrate the choices of 'Corridor', 'room', etc, the pit or even some of the monsters! Let us know if you try it.

PROGRAM LISTING

```
10L.
 1 REM  CELLS AND SERPENTS
 2 REM
 3 REM  This BBC BASIC translation by
 4 REM  I G Nicholls, February 1983.
 5 REM
 6 REM
 7 REM  Move display down a line
 8 REM
10 *TV255
20 MODE7
30 DATA36,100,5,9,100,6,10,100,6,6,4,4,10,5
31 REM
32 REM  Display title page and
33 REM  initialise variables
34 REM
40 PROCtitle
41 REM
42 REM  Load a character from tape?
43 REM
50 PROCload
51 REM
52 REM  Start of main program loop
53 REM
60 REPEAT
70   PROCbuild_1:CLS
80   PROCmonster
90   IF A(1)=3 AND A(2)=3 AND A(3)=3 PROCpit:GOTO140
100  PROCoptions:PROCoperate_cell(1)
110  IF RND(12)<>1 THEN GOTO130 ELSE PROCmonster:CLS:P
RINT'':PROCprint_monster:IF B$="a " OR B$="an " THEN PRIN
TB$;
120  PRINTA$;" arrives":R1=2:RM=1:PROCoperate_cell(2)
130  V1=0:RM=0
140  UNTIL FALSE
141 REM
142 REM  End of main program loop
143 REM
144 REM
145 REM
146 REM  You have fallen down a pit!
147 REM
150 DEF PROCpit
160 H=RND(12)
170 PRINT'"You fell down a ";H*10;" foot pit"
180 PRINT"you took ";:D1=RND(6)*H:PRINT;D1;" hps damage
"
190 IF RND(6)=1 GOTO210
200 H1=H1-D1:L1=L1+1:PROCdelay(20):PROCstatus_1:ENDPROC
210 PRINT"At the bottom there ";:H1=H1-D1
220 R=RND(3)
230 IF R=1 PRINT"is a pool of acid"'"you take ";:D1=RND
(8):PRINT;D1;" hps of damage":GOTO200
240 IF R=2 PRINT"are some spikes":R=RND(8):PRINT;R;" of
 which you hit":D1=R*RND(4):PRINT"doing ";D1;" hps of dam
age":GOTO200
250 PROCmonster:PROCprint_monster:PROCsuper_monsters:PR
INT"was"'B$;A$':L1=L1+1:PROCdelay(20):PROCcombat:ENDPROC
251 REM
252 REM  Increase hit-points for the really
253 REM  nasty monsters
254 REM
260 DEF PROCsuper_monsters
270 IF T1>80000 D1=D1+INT(4*(D1*(L1-1))↑0.5)
280 ENDPROC
281 REM
282 REM  Variable delay routine
283 REM
290 DEF PROCdelay(I%)
300 FOR Z%=0 TO I%*1000
310   NEXT
320 ENDPROC
321 REM
322 REM  Choose the noises heard behind the door
323 REM
330 DEF PROCmonster_noises:RN=RND(4):ON RN GOTO340,350,
360,370
340 PRINT"s snortins":ENDPROC
350 PRINT"s":ENDPROC
360 PRINT"s growlins":ENDPROC
370 PRINT"s howlins":ENDPROC
371 REM
372 REM  Build the next 3 choices
373 REM
380 DEF PROCbuild_1
390 FOR D1=1TO3
400   A1=RND(7)
410   R=RND(100)
420   IF A1<5 A(D1)=A1
430   IF L1=1 AND (A(1)=3 OR A(1)=5) AND (A(2)=3 OR A(2
)=5) A(3)=1:GOTO490
440   IF A1=5 IF R<10 A(D1)=A1:GOTO490
450   IF A1=6 IF R<25 A(D1)=A1:GOTO490
460   IF A1=7 IF R=1 A(D1)=A1:GOTO490
470   IF A1>4 A1=RND(4):A(D1)=A1
480   REM**NO GO HERE
490   NEXT
500 FOR D1=1TO3
510   IF A(D1)=2 PROCbuild_2
520   IF A(D1)=4 PROCbuild_2
530   NEXT
540 ENDPROC
541 REM
542 REM  Determine the room contents
543 REM
550 DEF PROCbuild_2
560 A(D1+3)=RND(4)
570 IF A(D1+3)=1 ENDPROC
580 IF A(D1+3)=4 ENDPROC
590 PROCmonster
600 FOR X=1TO14
610   A(-30*(D1=1)-45*(D1=2)-60*(D1=3)+X-1)=A(9+X)
620   NEXT
630 ENDPROC
631 REM
632 REM  Display your status
633 REM
640 DEF PROCstatus_1
650 PRINTCHR$135;CHR$157'CHR$135;CHR$157;CHR$132"Your h
it points stand at: ";H1
660 PRINTCHR$135;CHR$157;CHR$132"You have ";S1;" spells
"
670 PRINTCHR$135;CHR$157;CHR$132"You are on level: ";L1
680 PRINTCHR$135;CHR$157;CHR$132"You have ";G1;" gold p
ieces"
```

CONTINUED OVER

PROGRAM LISTING

```
  690 PRINTCHR$135;CHR$157;CHR$132"and you have killed ";
M1;" monsters!!"
  700 IF J1=0 PRINTCHR$135;CHR$157:GOTO720
  710 PRINTCHR$135;CHR$157;CHR$132"You also have a ";J1*2
5;"% luckstone"'CHR$135;CHR$157
  720 PRINTCHR$135;CHR$157'CHR$135;CHR$157;CHR$133;CHR$14
1;"Your score now is ";INT(SC)'CHR$135;CHR$157;CHR$133;CH
R$141;"Your score now is ";INT(SC)'CHR$135;CHR$157
  730 PROCstatus_2:ENDPROC
  731 REM
  732 REM  Show the 3 choices
  733 REM
  740 DEF PROCoptions
  750 PRINT''SPC(5);"LEFT";SPC(7);"FORWARDS";SPC(6);"RIGH
T"
  760 FOR D1=1TO3
  770   PRINTSPC(1);
  780   IF A(D1)=1 PRINTCHR$130;" Corridor  ";
  790   IF A(D1)=2 PRINTCHR$130;"   Door    ";
  800   IF A(D1)=3 PRINTCHR$129;"Blank wall ";
  810   IF A(D1)=4 PRINTCHR$130;"   Room    ";
  820   IF A(D1)=5 PRINTCHR$130;" Stairs up ";
  830   IF A(D1)=6 PRINTCHR$130;"Stairs down";
  840   IF A(D1)=7 PRINTCHR$136;CHR$131"  Exit   ";CHR$13
7;
  850   NEXT
  860 ENDPROC
  861 REM
  862 REM  Calculate the consequences of the choice made
  863 REM  This is the heart of the program
  864 REM
  870 DEF PROCoperate_cell(I%)
  880 ON I% GOTO890,1500
  890 PRINT''"What direction - L,F,R or S(status)?";:SS=0
  900 L=1:F=2:R=3:S=4
  910 PROCread_keyboard:D1=key
  920 D1=-((D1=76)+2*(D1=70)+3*(D1=82)+4*(D1=83))
  930 IF D1=S PROCstatus_1:GOTO890
  940 IF D1<>0 GOTO960
  950 PRINTCHR$11;CHR$11;CHR$11;CHR$11:GOTO890
  960 IF A(D1+3)<>2 IF A(D1+3)<>3 GOTO1000
  970 FOR X=1TO14
  980   A(9+X)=A(-30*(D1=1)-45*(D1=2)-60*(D1=3)+X-1)
  990   NEXT
 1000 IF A(D1)=7 PROCexit(1) ELSE ON A(D1) GOTO1010,1020,
1250,1330,1620,1660
 1010 ENDPROC
 1020 PRINT''"Door...O-open, L-listen ";
 1030 O=1:L=2:S=3
 1040 PROCread_keyboard:A=key
 1050 A=-((A=79)+2*(A=76)+3*(A=83))
 1060 IF A=S PROCstatus_1:GOTO1020
 1070 IF A<>0 GOTO1090
 1080 PRINTCHR$11;CHR$11;CHR$11:GOTO1020
 1090 IF A=O GOTO1330
 1100 IF A=L R1=A(D1+3)
 1110 PRINT''"You hear ";
 1120 IF R1<>2 AND R1<>3 PRINT"nothing":GOTO1160
 1130 R=RND(6)
 1140 IF R>2 PRINT"nothing":GOTO1160
 1150 PROCprint_monster:PRINT;A$;:PROCmonster_noises
 1160 PRINT"Do you want to open it? ";
 1170 Y=1:N=2:S=3
 1180 PROCread_keyboard:A=key
 1190 A=-((A=89)+2*(A=78)+3*(A=83))
 1200 IF A=S PROCstatus_1:GOTO1160
 1210 IF A<>0 GOTO1230
 1220 PRINTCHR$11;CHR$11:GOTO1160
 1230 IF A=Y GOTO1340
 1240 PRINT"...Chicken":PROCoptions:GOTO890
 1250 CLS:PRINT'CHR$136;CHR$131;SPC(5)"You can't move the
re dummy!"
 1260 IF RND(6)=1 PROCoptions:GOTO890
 1270 PRINT''"but as you have a liking for walls...."
 1280 PROCdelay(20)
 1290 PRINT"it falls over, you take ";:D1=RND(20)
 1300 PRINT;D1;" hps damage"
 1310 H1=H1-D1
 1320 PROCstatus_1:ENDPROC
 1330 R1=A(D1+3)
 1340 CLS:PRINT''"The room ";
 1350 IF R1=1 PRINT"is empty":PROCdelay(8):ENDPROC
 1360 PRINT"contains ";
 1370 IF R1=2 PROCprint_monster:PRINT;B$;A$;
 1380 IF R1=3 PRINT"treasure"''"+ ";:PROCprint_monster:PRI
NT;B$;A$;
 1390 IF R1=4 PRINT"treasure":T1=500:T=0
 1400 PRINT''"What now?"''"G-go in,L-leave,H-help,S-status
";
 1410 G=1:L=2:H=3:S=4
 1420 PROCread_keyboard:A=key
 1430 A=-((A=71)+2*(A=76)+3*(A=72)+4*(A=83))
 1440 IF A=S PROCstatus_1:GOTO1400
 1450 IF A<>0 GOTO1470
 1460 PRINTCHR$11;CHR$11;CHR$11;CHR$11;CHR$11:GOTO1400
 1470 IF A=L GOTO1240
 1480 IF A=H AND R1<4 PRINT"The ";:PROCprint_monster:PRIN
T;A$;" has ";D1;" hps":GOTO1400
 1490 IF R1=4 PROCtreasure(1):ENDPROC
 1500 IF SS=2 SS=0:PROCdelay(20):CLS
 1510 PRINT''"Now what?"''"C - combat,S - spell cast,R - re
treat ";
 1520 C=1:S=2:R=3
 1530 PROCread_keyboard:A=key
 1540 A=-((A=67)+2*(A=83)+3*(A=82))
 1550 IF A=R GOTO1590
 1560 IF A=C PROCcombat:ENDPROC
 1570 IF A=S IF S1<1 PRINT"Er..you don't seem to have any
":GOTO1500 ELSE PROCcast_spell:IF F%=1 GOTO1500 ELSE ENDP
ROC
 1580 PRINTCHR$11;CHR$11;CHR$11;CHR$11:GOTO1500
 1590 IF RND(6)=1 PRINT"TOUGH LUCK. He attacks you":PROCd
elay(8):PROCcombat:ENDPROC
 1600 IF RM=1 CLS:PRINT''"With a burst of cowardly speed,
...":PROCdelay(8):ENDPROC
 1610 GOTO1240
 1620 L1=L1-1
 1630 IF L1>0 ENDPROC
 1640 PRINT'''CHR$134;CHR$136;SPC(2);"Sorry, the stairs a
re blocked":L1=L1+1:PROCdelay(20)
 1650 CLS:PRINT':PROCoptions:GOTO890
 1660 L1=L1+1:ENDPROC
 1661 REM
 1662 REM  The EXIT - you've made it!
 1663 REM
 1670 DEF PROCexit(I%)
```

```
1680 ON I% GOTO1690,1700
1690 CLS:PRINTCHR$135;CHR$157'CHR$135;CHR$157;CHR$136;CH
R$133"Well done, you got out alive"'CHR$135;CHR$157
1700 E=1:PROCstatus_1:ENDPROC
1710 END:ENDPROC
1711 REM
1712 REM  Cast a spell and see if it works
1713 REM
1720 DEF PROCcast_spell
1730 F%=0
1740 S1=S1-1:SS=SS+1
1750 PRINT'"The now spell-blasted ";:PROCprint_monster:P
RINT;A$;
1760 IF A$="Intellect devourer" PRINTCHR$11
1770 H=RND(21)
1780 IF V1<>0 D1=V1
1790 D1=D1-H
1800 IF D1<1 PRINT'"lies dead on the floor":M1=M1+1:SC=S
C+L1:PROCtreasure(2):ENDPROC
1810 IF D1>0 PRINT'"is angry, he advances";:PROCdelay(8)
1820 IF RND(6)<3 PRINT",,he attacks":PROCdelay(20):PROCc
ombat:ENDPROC
1830 V1=D1:F%=1:ENDPROC
1831 REM
1832 REM  This is where you fight the monster,
1833 REM  and see who wins
1834 REM
1840 DEF PROCcombat
1850 CLS:PRINT'"You had a terrific battle with ";:PROCpr
int_monster:IF B$="a " OR B$="an " PRINT"the"
1860 PRINTA$;
1870 IF V1<>0 D1=V1
1880 IF H1>D1 PRINT" and you killed him":PROCdelay(20):M
1=M1+1:H1=H1-D1:SC=SC+L1:PROCtreasure(2):ENDPROC
1890 PRINT", but he killed you"'"and took all your treas
ure":G1=0:H1=H1-D1:PROCexit(2):ENDPROC
1891 REM
1892 REM  This routine checks to see if you:-
1893 REM
1894 REM  - are dead
1895 REM  - have found the exit
1896 REM  - want to continue
1897 REM  - want to save your character on tape
1898 REM
1900 DEF PROCstatus_2
1910 Z$=CHR$135+CHR$157+CHR$129:IF H1<=0 THEN PROCdead:E
NDPROC
1920 IF E<>1 THEN 1980
1930 IF H1<150 H1=150
1940 IF S1<15 S1=15
1950 L1=1:E=0:PRINTZ$'Z$'Z$;"Having left the catacombs,
you can"
1960 PRINTZ$;"only enter again at level 1 but your"
1970 PRINTZ$;"hps and spells have been replenished"'Z$
1980 PRINTZ$'Z$;"Press space to continue"
1990 PRINTZ$;"Press Q to save character on tape";
2000 *FX15,1
2010 z$=GET$:IF ASC(z$)<>32 AND z$<>"Q" PRINTCHR$11:GOTO
1990
2020 IF z$<>"Q" ENDPROC ELSE PROCsave_character:END:ENDP
ROC
2021 REM
2022 REM  Save your character on tape,
2023 REM  and leave the program
2024 REM
2030 DEF PROCsave_character
2040 INPUT'"What is your name,"'"(maximum of 10 charact
ers) ",NM$
2050 IF LEN(NM$)>10 PRINT:GOTO2030
2060 PRINT'"INSERT AND REWIND DATA TAPE"
2070 PRINT'"Press any key when you are ready"'"to recor
d data"'
2080 ZZ=GET
2090 XY=OPENOUT(NM$)
2100 PRINT#XY,H1,S1,L1,G1,M1,J1,SC
2110 CLOSE#XY
2120 ENDPROC
2121 REM
2122 REM  Read a character from tape
2123 REM
2130 DEF PROCread_character
2140 INPUT'"What is your name,"'"(maximum of 10 charact
ers) ",NM$
2150 IF LEN(NM$)>10 PRINT"too long...."':GOTO2130
2160 PRINT'"INSERT AND REWIND DATA TAPE"
2170 PRINT'"Press any key to load data"'
2180 ZZ=GET
2190 XY=OPENIN(NM$)
2200 INPUT#XY,H1,S1,L1,G1,M1,J1,SC
2210 CLOSE#XY
2220 ENDPROC
2221 REM
2222 REM  You are dead but you may play again
2223 REM
2230 DEF PROCdead
2240 PRINT'"Do you wish to play again?";
2250 W$=GET$:IF W$<>"Y" AND W$<>"N" PRINTCHR$11:GOTO2230
2260 IF W$="Y" RUN
2270 END:ENDPROC
2271 REM
2272 REM  What treasure lies inside the room?
2273 REM
2280 DEF PROCtreasure(I%)
2290 ON I% GOTO2300,2310
2300 IF T1=0 PRINT"It was an illusion":PROCdelay(8):ENDP
ROC
2310 IF R1=2 PROCdelay(8):ENDPROC
2320 PRINT"There are:-"'
2330 R=INT(LOG((RND(1)*T1+10)*INT(RND(1)*L1/2+1))*2)
2340 IF J1>0 R=R+INT(((25*J1)/100)*R)
2350 PRINT;R;" gold pieces";:SC=SC+R/100:G1=G1+R
2360 IF T=0 AND RND(100)<15+L1*2 GOTO2380
2370 IF T=0 PROCdelay(8):ENDPROC
2380 FOR X=1TO T
2390   R=RND(10)
2400   IF R=1 PRINT" + a sword":H1=H1+RND(120)
2410   IF R=2 PRINT" + a wand":S1=S1+RND(15)
2420   IF R=3 PRINT" + a suit of armour":H1=H1+RND(90)
2430   IF R=4 PRINT" + a scroll":S1=S1+RND(12)
2440   IF R=5 PRINT" + some more spells":S1=S1+RND(8)
2450   IF R=6 PRINT" + a potion":H1=H1+RND(75)
2460   IF R=7 PRINT" + a special artefact":H1=H1+RND(100
):S1=S1+RND(12)
2470   IF R=8 PRINT" + a book":PROCrandom
2480   IF R=9 PRINT" + a ring":PROCrandom
2490   IF R=10 Z=RND(6)/2
2500   IF R=10 AND Z>J1 PRINT" + a ";Z*25;"% luckstone"
```

CONTINUED OVER

PROGRAM LISTING

```
2510   IF R=10 AND J1<Z J1=Z
2520   IF RND(10)=1 GOTO2390
2530   NEXT
2540 PROCdelay(20):ENDPROC
2541 REM
2542 REM  If the treasure is a book or a ring then this
2543 REM  routine calculates your hit-points and spells
2544 REM
2550 DEF PROCrandom
2560 IF RND(2)=1 H1=H1+RND(80):PROCdelay(20):ENDPROC
2570 S1=S1+RND(10):PROCdelay(20):ENDPROC
2571 REM
2572 REM  Choose a monster at random
2573 REM
2580 DEF PROCmonster
2590 RESTORE
2600 FOR Z=1TO14
2610   READ D:A(9+Z)=RND(D)
2620   NEXT
2630 ENDPROC
2631 REM
2632 REM  Put the monster's name into A$
2633 REM  and "a" or "an" into B$
2634 REM  Also determine the monster's hit-points (D1),
2635 REM  and "treasure" points (T1), used to work out
2636 REM  the number of gold pieces in the room
2637 REM
2640 DEF PROCprint_monster
2650 R=A(10)
2660 P=A(11):T=0
2670 IF R=1 A$="Wraith":B$="a ":D1=10:T1=8000:ENDPROC
2680 IF R=2 A$="Vampire":B$="a ":D1=20:T1=10000:ENDPROC
2690 IF R=3 AND P<65 PROCserpents:A$=A$+"Serpent":ENDPRO
C
2700 IF R=4 AND P<65 PROCdemons:ENDPROC
2710 IF R=5 AND P<65 PROCdevils:ENDPROC
2720 IF R=6 A$="Troll":B$="a ":D1=16:T1=6000:ENDPROC
2730 IF R=7 A$="Kobold":B$="a ":D1=1:T1=300:ENDPROC
2740 IF R=8 A$="Ghost":B$="a ":D1=32:T1=8000:T=1:ENDPROC
2750 IF R=9 PROCgiants:A$=A$+"Giant":ENDPROC
2760 IF R=10 A$="Hydra":B$="a ":D1=50:T1=4000:ENDPROC
2770 IF R=11 AND P<65 A$="Intellect devourer":B$="an ":D
1=20:T1=6000:ENDPROC
2780 IF R=12 A$="Salamander":B$="a ":D1=20:T1=9000:T=2:E
NDPROC
2790 IF R=13 A$="Zombie":B$="a ":D1=8:T1=0:ENDPROC
2800 IF R=14 A$="Aerial servant":B$="an ":D1=40:T1=0:END
PROC
2810 IF R=15 A$="Basilisk":B$="a ":D1=10:T1=6000:T=1:END
PROC
2820 IF R=16 A$="Beholder":B$="a ":D1=42:T1=15000:T=3:EN
DPROC
2830 IF R=17 AND P<65 PROCelementals:A$=A$+"Elemental":T
1=0:ENDPROC
2840 IF R=18 A$="Ettin":B$="an ":D1=39:T1=12000:ENDPROC
2850 IF R=19 A$="Gargoyle":B$="a ":D1=10:T1=1000:ENDPROC
2860 IF R=20 A$="Giant wolf":B$="a ":D1=34:T1=12000:ENDP
ROC
2870 IF R=21 AND P<65 PROCgolems:A$=A$+"Golem":T1=0:ENDP
ROC
2880 IF R=22 A$="Hell hound":B$="a ":D1=12:T1=1000:ENDPR
OC
2890 IF R=23 AND P<65 PROClich:ENDPROC
2900 IF R=24 A$="Were-":B$="a ":PROClycanthropes:ENDPROC
2910 IF R=25 A$="Manticore":B$="a ":D1=48:T1=8000:T=1:EN
DPROC
2920 IF R=26 A$="Medusa":B$="a ":D1=15:T1=12000:T=1:ENDP
ROC
2930 IF R=27 AND P<65 A$="Mind flayer":B$="a ":D1=48:T1=
4000:T=2:ENDPROC
2940 IF R=28 A$="Minotaur":B$="a ":D1=10:T1=5000:ENDPROC
2950 IF R=29 A$="Mummy":B$="a ":D1=12:T1=5000:ENDPROC
2960 IF R=30 A$="Orc":B$="an ":D1=2:T1=500:ENDPROC
2970 IF R=31 A$="Purple worm":B$="a ":D1=56:T1=9000:T=2:
ENDPROC
2980 IF R=32 AND P<65 A$="Umber hulk":B$="an ":D1=34:T1=
40000:T=2:ENDPROC
2990 IF R=33 A$="Wight":B$="a ":D1=8:T1=8000:ENDPROC
3000 IF R=34 AND P<65 A$="Xorn":B$="a ":D1=34:T1=40000:T
=2:ENDPROC
3010 A$="Hobsoblin":B$="a ":D1=6:T1=1000:ENDPROC
3011 REM
3012 REM  Choose a serpent
3013 REM
3020 DEF PROCserpents
3030 R=A(12)
3040 B$="a "
3050 IF R=1 A$="Black ":D1=50:T1=60000:T=2:ENDPROC
3060 IF R=2 A$="White ":D1=40:T1=50000:T=1:ENDPROC
3070 IF R=3 A$="Blue ":D1=60:T1=70000:T=2:ENDPROC
3080 IF R=4 A$="Green ":D1=70:T1=80000:T=3:ENDPROC
3090 A$="Red ":D1=80:T1=90000:T=4:ENDPROC
3091 REM
3092 REM  Choose a demon
3093 REM
3100 DEF PROCdemons
3110 R=A(13)
3120 P=A(14)
3130 IF R=1 AND P<5 A$="Demosorson":B$=" ":D1=95:T1=1000
00:T=6:ENDPROC
3140 IF R=2 AND P<5 A$="Jubilex":B$=" ":D1=80:T1=81000:T
=2:ENDPROC
3150 IF R=3 AND P<5 A$="Orcus":B$=" ":D1=105:T1=150000:T
=7:ENDPROC
3160 A$="Type ":B$="a "
3170 IF R=4 A$=A$+"VI":D1=60:T1=60000:T=3:GOTO3240
3180 IF R=5 A$=A$+"V":D1=50:T1=50000:T=2:GOTO3240
3190 IF R=6 A$=A$+"IV":D1=40:T1=40000:T=1:GOTO3240
3200 IF R=7 A$=A$+"III":D1=30:T1=30000:GOTO3240
3210 IF R=8 A$=A$+"II":D1=20:T1=20000:GOTO3240
3220 IF R=9 A$=A$+"I":D1=10:T1=10000:GOTO3240
3230 IF R<=3 R=A(15)+3:GOTO3170
3240 A$=A$+" DEMON":ENDPROC
3241 REM
3242 REM  Choose a devil
3243 REM
3250 DEF PROCdevils
3260 R=A(16)
3270 P=A(17)
3280 IF R=1 AND P<5 A$="Asmodeus":B$=" ":D1=110:T1=17000
0:ENDPROC
3290 IF R=2 AND P<5 A$="Baalzebul":B$=" ":D1=80:T1=81000
:T=5:ENDPROC
3300 IF R=3 AND P<5 A$="Dispater":B$=" ":D1=70:T1=81000:
T=3:ENDPROC
3310 IF R=4 AND P<5 A$="Geryon":B$=" ":D1=50:T1=81000:T=
2:ENDPROC
```


```
3320 IF R=5 A$="Barbed Devil":B$="a ":D1=32:T1=0:ENDPROC
3330 IF R=6 A$="Bone Devil":B$="a ":D1=35:T1=0:ENDPROC
3340 IF R=7 A$="Erinyes":B$="an ":D1=16:T1=40000:ENDPROC
3350 IF R=8 A$="Horned Devil":B$="a ":D1=35:T1=5000:ENDP
ROC
3360 IF R=9 A$="Ice Devil":B$="an ":D1=60:T1=10000:T=3:E
NDPROC
3370 IF R=10 A$="Pit fiend":B$="a ":D1=65:T1=12000:T=4:E
NDPROC
3380 R=A(18)+4:GOTO3320
3381 REM
3382 REM  Choose a giant
3383 REM
3390 DEF PROCgiants
3400 R=A(19)
3410 B$="a "
3420 IF R=1 A$="Cloud ":D1=36:T1=9000:ENDPROC
3430 IF R=2 A$="Fire ":D1=30:T1=8000:ENDPROC
3440 IF R=3 A$="Frost ":D1=24:T1=8000:ENDPROC
3450 IF R=4 A$="Hill ":D1=16:T1=3000:ENDPROC
3460 IF R=5 A$="Stone ":D1=18:T1=4000:ENDPROC
3470 A$="Storm ":D1=42:T1=10000:T=1:ENDPROC
3471 REM
3472 REM  Choose an elemental
3473 REM
3480 DEF PROCelementals
3490 R=A(20)
3500 IF R=1 A$="Air ":B$="an ":D1=90:ENDPROC
3510 IF R=2 A$="Earth ":B$="an ":D1=65:ENDPROC
3520 IF R=3 A$="Fire ":B$="a ":D1=48:ENDPROC
3530 A$="Water ":B$="a ":D1=60:ENDPROC
3531 REM
3532 REM  Choose a golem
3533 REM
3540 DEF PROCgolems
3550 R=A(21)
3560 IF R=1 A$="Clay ":B$="a ":D1=30:ENDPROC
3570 IF R=2 A$="Flesh ":B$="a ":D1=32:ENDPROC
3580 IF R=3 A$="Iron ":B$="an ":D1=90:ENDPROC
3590 A$="Stone ":B$="a ":D1=24:ENDPROC
3591 REM
3592 REM  Choose a hobgoblin or a lich
3593 REM
3600 DEF PROClich
3610 D=A(22)
3620 B$="a "
3630 IF D>6 A$="Hobgoblin":D1=6:T1=1000:ENDPROC
3640 A$="Lich":D1=90:T1=80000:T=5:ENDPROC
3641 REM
3642 REM  Choose a were-creature
3643 REM
3650 DEF PROClycanthropes
3660 R=A(23)
3670 IF R=1 A$=A$+"bear":D1=10:T1=2500:ENDPROC
3680 IF R=2 A$=A$+"boar":D1=12:T1=3000:ENDPROC
3690 IF R=3 A$=A$+"rat":D1=8:T1=2000:ENDPROC
3700 IF R=4 A$=A$+"tiger":D1=20:T1=5000:ENDPROC
3710 A$=A$+"wolf":D1=8:T1=2000:ENDPROC
3711 REM
3712 REM  Display the title and initialise
3713 REM
3720 DEF PROCtitle
3730 DIM A(80)
3740 L1=1:G1=0:H1=100:M1=0:J1=0:S1=10:T=0:D1=1:V1=0:SC=0
:E=0
3750 Z%=RND(-TIME)
3760 VDU23,0,11,0,0,0,0,0,0,0
3770 FOR I%=0TO24:PRINTTAB(0,I%)CHR$135;CHR$157;
3780   NEXT
3790 PRINTTAB(13,8);CHR$141;CHR$132;"CELLS  AND";TAB(13,
9);CHR$141;CHR$132;"CELLS  AND"'
3800 PRINTTAB(14,12);CHR$141;CHR$132;"SERPENTS";TAB(14,1
3);CHR$141;CHR$132;"SERPENTS"
3810 PROCdelay(30)
3820 VDU23,0,11,255,0,0,0,0,0,0
3830 CLS
3840 ENDPROC
3841 REM
3842 REM  Character load routine
3843 REM
3850 DEF PROCload
3860 PRINT''"Do you wish to load a character"
3870 PRINT"from tape?";
3880 W$=GET$:IF W$<>"Y" AND W$<>"N" PRINTCHR$11:GOTO3870
3890 IF W$="Y" PROCread_character
3900 ENDPROC
3901 REM
3902 REM  Read a single key depression
3903 REM  and print the character
3904 REM
3910 DEF PROCread_keyboard
3920 *FX15,1
3930 Key=GET
3940 PRINTCHR$(Key)
3950 ENDPROC
```

PROGRAM LISTING

```
100 REM  Patch for 2 bugs in cassette filing
101 REM  system. Only needed for v0.1 of MOS
102 REM  DO NOT USE WITH OTHER VERSIONS
130 FORPASS=0TO1:P%=&UDO:COSUB180:NEXT
140 ?&218=FIX1:?&219=FIX1 DIV 256
150 ?&20A=FIX2:?&20B=FIX2 DIV 256
160 *KEY10 ?&218=&DO:?&219=&D:?&20A=&D6:?&20B=&DIN
165 CHAIN""
170 END
180 [OPT PASS*2
190 .FIX1 PHA:JSR &F521:PLA:RTS
200 .FIX2 CMP #&91:BNE GO:CPX #0:BNE GO
210 TSX:LDA &102,X:CMP #&F7:BEQ TRAP
220 LDX #0:.IX LDA #&91:STA &FE09:RTS
230 .GO JMP (&DB60)
240 .TRAP PLA:PLA
250 JSR &F9D8:JSR &FB7B
260 JSR TX:JMP &F7FB
270 ]RETURN
```

The program patch required if you possess version 0.1 of the Machine Operating System. It is not required otherwise.

In the Beginning

G W Gallagher

This is the first part of our regular 'Learning BASIC' series, taking you from the first steps of programming.

During the last few months, BBC Micros have arrived in hundreds of schools, and not every school has a member of staff brave enough to take on the new challenge. It is a challenge, partly because of the widely-held, and incorrect, belief that it is necessary to have some extra mathematical ability to use a computer.

NEWTONS THEORY

Mathematicians and scientists perhaps gravitate towards the computer more naturally, because they can see immediate and obvious uses for it in diagrams and calculations. But, in the last few months, I have seen how historians and geographers, teachers of English and foreign languages, and geologists thrived when they have undertaken a regular session at a computer keyboard, trying to familiarise themselves with the monster which has inspired such enthusiasm in pupils and in their owns sons and daughters. In most cases, the task was undertaken with trepidation, but continued with enjoyment!

This series is therefore addressed to those of you who, until recently would not even have considered looking at a computer, but are beginning to realise that, in 1983, they are objects which cannot be avoided at home or at school.

COMMUNICATION BREAKDOWN

Communicating with a computer is a question of language, and, as with any other language, there is a vocabulary to be learned, and the structure of a grammar to be understood. However, there is one real difference—if you are trying to communicate with someone whose native language is not your own, there is usually some success in understanding the meaning of what is said, even if the actual words are difficult to understand.

With a computer, there is no such tolerance, It can only understand the language in which it has been programmed, and has no intuition to help it to understand what you are trying to say. In this case, the language of communication is BASIC.

If you have the computer in front of you, and it is attached to a monitor or television, then it is worth spending a few minutes looking at the keyboard. When the computer is switched on (the small switch at the back left-hand side), a small red light will appear under the heading 'CAPS LOCK'. As long as this light is on, the letter keys will produce capital letters on the screen. Press the Caps Lock key once, the light will go out, and the letters transferred to the screen will be lower case. Press the same key again, and the capital letters will once more be there.

The keyboard itself is much like a typewriter keyboard, as far as the letters and numbers are concerned. The red keys on the top row have their own use which will be described later. At the extreme right of the red keys is the Break key. When this key is pressed, the screen is cleared, and the memory is cleared also. (If you are writing a program, and press this key accidentally, type OLD immediately, and press Return. By the magic of the BBC Micro, your program is returned to you, unlike some other computers!)

If you type a message from the keyboard, the letters will immediately appear on the screen. The computer does not act upon any message until the Return key is pressed. In this case, when you pressed Return, the message 'Mistake' will almost certainly appear. It could not understand the message!

As each letter is typed, a flashing white line moves along screen, marking the next typing position. This is the Cursor. It can be moved around the screen by using the four direction arrows at the top right-hand side of the keyboard. (By now the screen is probably covered with letters or numbers). If the Cursor is moved up to any line, and the Copy key pressed, the line will be repeated at the place where the cursor was, a place not marked by a white square, which will move along the line as the cursor did.

The direction arrows, Copy and Delete keys are the means used to Edit lines, ie after lines already written. It is usually a quicker and simpler procedure than to type out the line again.

Type CLS and press Return, making the screen clear. Now type 2 + 5, and press Return. The computer will not respond with the answer, not because it cannot cope with the arithmetic but because it needs instruction as to where the answer should be placed.

Try typing:

```
PRINT 2 + 5
```

and press RETURN. The answer 7 will immediately appear on the screen. The 'PRINT' is the necessary instruction. (If you tire of typing PRINT, the abbreviation 'P.' will be accepted.) The other arithmetical signs which could be tried here are:

```
* multiply
/ divide
- subtract
  to the power of
```

Since the computer works in MODE 7, the teletext mode,

unless changed, the symbol ^ will appear on the screen as ↑ To see the difference, type MODE 5, and press Return. The screen will clear, and the letters become larger. Type:

```
PRINT 2^3
```

and press Return. The '^' will be on the screen as it appears on the keyboard. Now type MODE 7 and try the same calculation again. This time there will be an '↑' instead of '^'.

The computer has 'remembered' none of the working carried out so far. It has provided instant answers when required, but to take any instruction into its memory, it must be approached by means of a Program.

WRITING A PROGRAM

A program may be composed of one line of hundreds of lines, but each line must begin with a number. The number of lines will decide the order in which the program is carried out, so that it is usual to number in 10s, to make allowances for lines being added later. Lines need not be entered in the correct numerical order when a program is being written, as the computer will automatically sort the lines into the correct numerical order.

If we had written:

```
10 PRINT 2+5
```

and pressed Return, the single line would have been stored in the memory as a program. For the addition to be carried out, type RUN and press Return.

At this stage, it is worth emphasising that the computer cannot carry out an instruction or enter a program line until the *Return key has been pressed.*

This line will now stay in the memory, and the program may be run at any time, unless one of four events occur.

1. Typing 10 and pressing the Return key removes line 10 from the memory.
2. Typing a new line 10 removes the old one.
3. Typing NEW (and pressing the Return key) clears the old program from memory.
4. Pressing the Break key clears the memory (unless OLD is typed immediately as described earlier).

Use one of these methods to clear this short program from memory and start again. Because we usually intend to use more than one set of numbers, variable names may be used instead of actual values. Variable names may be of any length, but must not include one of BASIC language variables (see pages 483-4 of the **User Guide**).

Try the following program:

```
↑LIST
  20 PRINT"What is the value of A?"
  30 INPUT A
  40 PRINT"What is the value of B?"
  50 INPUT B
  60 PRINT A+B
```

When the program is typed in, type RUN (and press Return). If there is an error in the typing, the screen will show an error message and a line number. Typing LIST (and Return) will list the whole program. Typing LIST 40 will list just line 40. Check the line to find the mistake, and use the cursor controls and COPY to correct it. When the program is RUN, the message,

What is the value of A?

will appear on the screen, and the cursor will wait, flashing, until a number has been typed in. This number will become the value of A, and the second number asked for will become B, giving the sum as the answer.

To feed in another pair of numbers, the program must be RUN again, which becomes rather frustrating. We can illustrate two ways of overcoming this (there are others!). Type in the extra line:

```
70 GOTO 20
```

and RUN again. Now, each time the program causes an answer to be printed out, line 70 will send it back to the beginning again. (It is important that 'GOTO' be typed as one word, otherwise the computer will not recognise it.) This program will continue indefinitely and it is necessary to press the Escape key to get out of it.

CONTINUED OVER

For a different method, type in the following lines:

```
10  FOR N=1 TO 5
70  NEXT N
```

When the program is listed it now looks like:

```
>LIST
  10  FOR N=1 TO 5
  20    PRINT"What is the value of A?"
  30    INPUT A
  40    PRINT"What is the value of B?"
  50    INPUT B
  60    PRINT A+B
  70    NEXT N
```

Lines 10 and 70 then form a loop, and the letter N (or any other variable) is used as a counter. In this example, every line in the loop, ie between lines 10 and 70, is repeated 5 times, as will become apparent when the program is RUN.

So far we have only used numbers, but many programs require letters as well as numbers as input. Any variable which is not a number is a string variable, and its variable name must end with a dollar sign, eg A$, B$, NAME$.

Alter the program already in memory until it reads:

```
>LIST
  10  FOR N=1 TO 5
  20    PRINT"Christian name?"
  30    INPUT A$
  40    PRINT"Surname?"
  50    INPUT B$
  60    PRINT A$ +" "+B$
  70    NEXT N
```

Now, when the program is RUN, the computer will expect strings of letters, or a mixture of letters and numbers. It will accept all numbers but will not recognise them as such, so they cannot be used in that form in arithmetical calculations.

Line 60 prints the Christian name, followed by a space and then the surname.

USING STRINGS

Strings are a very useful tool in writing a program as they can be manipulated and split into sections as required. Useful string functions are:

LEFT$(A$,4) which would use the four character from the left of the string A$.

RIGHTS(A$,4) which would read the last four characters from the right of the string A$.

MID$(A$,2,4) which would take four characters, starting with the second from the left of the string A$.

STRING$(6,A$) which would repeat the whole of the string A$ six times (Beware—the maximum length of the string produced is 256 characters).

Try the following illustration:

```
>LIST
  10 A$="GEOGRAPHICAL"
  20 B$="HISTORICAL"
  30 C$="ASTRONOMICAL"
  40 FOR T=1 TO 12
  50   PRINT LEFT$(A$,T)
  60   NEXT T
  80 FORT=1 TO 10
  90     PRINT RIGHT$(B$,T)
 100   NEXT T
 120 FORT=1 TO 9
 130   PRINT MID$(C$,T,4)
 140   NEXT T
 150 END
```

When the program is run, it will flash quickly through the screen, and the first part is lost to view before there has been the opportunity to read it. This suggests that some kind of delaying device is needed. This could be done with a subroutine of a PROC. Leaving the introduction to PROCs until later, the following lines will allow the program to move at the operator's request.

```
200   PRINT "PRESS ANY KEY TO CONTINUE"
210   X$=GET$:IF X$="" THEN 210
220   RETURN
```

GET$ is an instruction to the computer to take in any key which has been pressed. If no key is pressed then the program goes back to the beginning of line 210, and continues to circle there until a key is pressed. 'Any key' really means any of the letter or number keys, or the Space Bar.

To include this routine in the program, instructions must be inserted in the program to send it to line 200 when a delay is wanted.

```
 70  GOSUB 200
110  GOSUB 200
```

Run the program again to see the complete effect.

In the next instalment, strings will be used to provide information to form the basis of tests.

```
>LIST
  10 A$="GEOGRAPHICAL"
  20 B$="HISTORICAL"
  30 C$="ASTRONOMICAL"
  40 FOR T=1 TO 12
  50   PRINT LEFT$(A$,T)
  60   NEXT T
  70  GOSUB200
  80 FORT=1 TO 10
  90     PRINT RIGHT$(B$,T)
 100   NEXT T
 110  GOSUB200
 120 FORT=1 TO 9
 130   PRINT MID$(C$,T,4)
 140   NEXT T
 150 END
 200  PRINT"PRESS ANY KEY TO CONTINUE"
 210  X$=GET$:IF X$="" THEN 210
 220  RETURN
```

```
>RUN
G
GE
GEO
GEOG
GEOGR
GEOGRA
GEOGRAP
GEOGRAPH
GEOGRAPHI
GEOGRAPHIC
GEOGRAPHICA
GEOGRAPHICAL
PRESS ANY KEY TO CONTINUE
L
AL
CAL
ICAL
RICAL
ORICAL
TORICAL
STORICAL
ISTORICAL
HISTORICAL
PRESS ANY KEY TO CONTINUE
ASTR
STRO
TRON
RONO
ONOM
NOMI
OMIC
MICA
ICAL
```

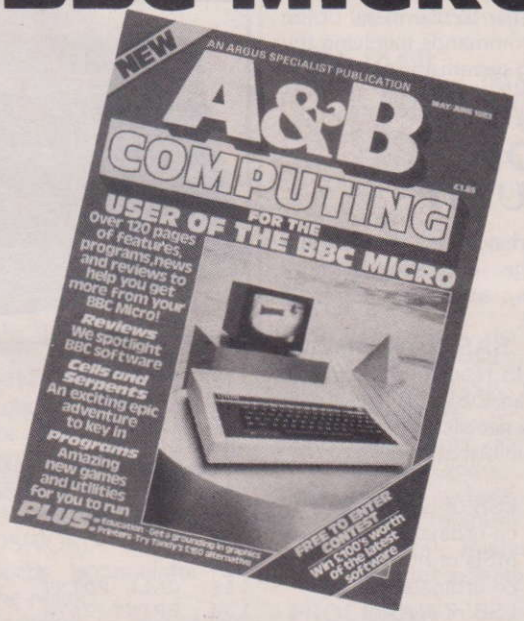
Schooldays: computers are now an everyday part of school life.

A&B COMPUTING FOR THE USER OF THE BBC MICRO

A&B Computing is a new bi-monthly publication dedicated to users of the BBC Micro.

Priced at £1.85, forthcoming issues of A&B Computing will contain features on various aspects of the system, courses on improving your techniques in graphics, sound and assembly language, reviews of software and new add-ons, as well as plenty of exciting games and utility programs for you to run. Indeed, everything you'll need to make the most of your BBC Micro.

So, no matter whether you're a raw beginner or an accomplished programmer, we're sure you'll find A&B Computing the magazine you simply can't do without. And, just to prove it, we're offering you the chance to take out a subscription at the same rates that you would pay in the shops and not a penny more.



DON'T MAKE YOUR BBC MICRO MISS OUT, SUBSCRIBE TO A&B COMPUTING TODAY!

Subscription Order Form

Cut out and SEND TO:

A&B COMPUTING
513 London Road
Thornton Heath
Surrey CR4 6AR

Please commence my subscription with the very next issue.

SUBSCRIPTION RATES (tick □ as appropriate)

£11.10 for six issues UK □
£13.40 for six issues overseas surface mail □
£21.00 for six issues overseas airmail □

I am enclosing my (delete as necessary) cheque/Postal Order/International Money Order for £.....
(made payable to ASP Ltd)
OR
Debit my Access/Barclaycard* (*delete as necessary)

BARCLAYCARD VISA

Please use BLOCK CAPITALS and include post codes.

NAME (Mr/Mrs/Miss) .....

ADDRESS .....

.....

..... POSTCODE .....

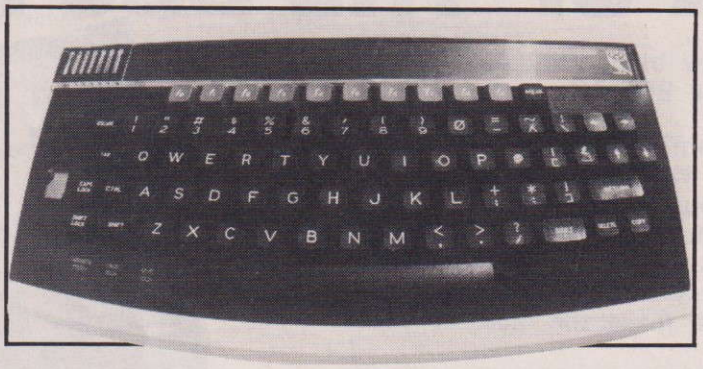
Signature .....

Date .....

Osword Calls

M Stanger

Speed up your BASIC programs with this clever operating system routine.



The operating system call, OSWORD, is mainly used in machine code progams and consists of a JSR to location &FFF7 Hex.

The call number is passed in the accumulator and the X and Y resisters are used to pass the first location of the parameter block containing the other values. This will enable the machine code programmer to use many of the BASIC commands involving the operating system and can be best illustrated using one or two examples.

READ ALL ABOUT IT

For the rest of this discussion a knowledge of Hex, memory locations, and indirection is assumed.

The BASIC command, POINT, to read the colour of a pixel is an OSWORD call with A=9. It is necessary to pass four bytes of information:

XY	LSB of horizontal co-ordinate.
XY+1	MSB of horizontal co-ordinate.
XY+2	LSB of vertical co-ordinate.
XY+3	MSB of vertical co-ordinate.

and to reserve one byte, XY+4, for the answer.

In the example to read the pixel (288,576), the parameter block is from &70 to &74 Hex on Page Zero. The horizontal and vertical co-ordinates are put in these locations using indirection. On entry, the X register contains the &70 Hex (LSB of start of parameter block) and the Y register contains 0 (HSB of start of parameter block). After the program has been called, the result is read from &74 Hex.

```
10  MODE 4
20  DRAW 1280,1023:DRAW
    0,1023:PLOT 85,0,0
30  !&70=288
40  !&72=576
50  OSWORD=&FFF1
60  [OPT0
70  .POINT
80  LDA#9
90  LDX#&70
100 LDY#0
110 JSR OSWORD
120 RTS
130 ]
140 CALL POINT
150 PRINT ?&74
160 END
```

The interval timers can be read and written with OSWORD calls three and four.

OSWORD call with A=4 writes the interval timer. On entry, X and Y point to five locations containing the new setting. In this example, the new setting (minus WAIT) is measured in hundredths of a second. The parameter block starts at &8B Hex. On entry, the X register contains the LSB and the Y register the MSB of &008B Hex.

The OSWORD call with A= reads the interval timer. In the following example, the time is set as above and the program will continue to read the time until the setting is zero and the delay is over.

```
10  PROCINIT
20  PROCDELAY(100)
30  VDU 7
40  END
50  DEFPROCINIT
60  OSWORD=&FFF1
70  DIM P%30
80  DELAY=&D00
90  [OPT 2
100 .DELAY
110 LDX#&8B
120 LDY#&0
130 LDA#4
140 JSR OSWORD
150 .STAY LDA#3
160 JSR OSWORD
170 LDA&8F
180 BMI STAY
190 RTS
200 ]
210 ENDPROC
220 DEFPROCDELAY(WAIT)
230 !&8B=-WAIT:?&8F=&FF
    :CALL DELAY
240 ENDPROC
```

The two examples above are used in a program to copy the high resolution screen on to an Amber 2400 printer.

The first routine reads the selected pixel, and the second program slows down the printing of dense graphics.

THE AMBER 2400 PRINTER

The width the printer can handle is 144 dots and a single byte from the computer can deal with any combination of eight dots so that a single line of dots in the graphics mode can be represented by 18 characters.

The graphics mode is turned on by the output, 17, after which the printer will store the next 18 characters in a line buffer ready to be printed. The speed of the BASIC program in the **User Guide** can be improved using machine code.

The BASIC program is shown here with a few alterations.

Lines Description

15 Selects a graphics mode.

16 Draws a triangle to test the program.

70 There is a misprint in line 70 in my copy of the **User Guide.** The upper limit of X1 should be X+28 rather than X+128 as there are eight graphics needles not 33. This alteration makes the BASIC program four times as fast.

80 Slight alteration to use the program in other modes.

The machine code program, SCREENDUMP, is faster than the BASIC program, but there is a short delay incorporated to double the printing time when graphics are dense.

```
10  REM ** PRINT SCREEN
    (0,0) TO (576,576)
15  MODE 4
16  DRAW 576,576:DRAW 0
    ,576:DRAW 0,0
20  *FX 5,1
30  FOR Y=576 TO 0
    STEP -4
40  A%=&11:VDU 1:PRINT
    CHR$(A%)
50  FOR X=0 TO 576
    STEP 32
60  A%=0:B%=128
70  FOR X1=X TO X+28
    STEP 4
80  IF POINT(X1,Y)<>0
    THEN A%=A%+B%
90  B%=0.5*B%
100 NEXT X1
110 VDU 1:PRINT CHR$(A%)
120 NEXT X:NEXT Y
130 END
```

BASIC PROGRAM VARIABLES

X1 Horizontal screen location.
Y Vertical screen location.
X Horizontal screen position of first graphics needle.
A% Character to be printed.
B% Needle position.

These variable names will also be used in the program description for SCREENDUMP.

SCREENDUMP VARIABLES

YL Lowest screen location printed.
YH Highest screen location printed.
P(A) and Q(A) Used to store the calculated positions of the vertices in the drawing procedure.

ZERO PAGE VARIABLES

&70 Low byte of X1.
&71 High byte of X1.
&72 Low byte of Y.
&73 High byte of Y.
&74 Number of lines to be printed (one byte). Later contains POINT (X1, Y).
&75 Low byte of X.
&76 High byte of X.
&80 A% (1 byte).
&81 B% (1 byte).
&82 Number of needles firing (one byte).
&8B to &8F Five bytes - (waiting time).

PROGRAM DESCRIPTION

This program draws a geometric shape on the screen and prints the screen locations from (0,0) to (576,576) on an Amber 2400 Printer.

Lines Description

90 Selects Mode 4 and removes the cursor.
100 Calls a procedure to dimension the vertices for the drawing and assembles the printing routine.
110 Calls a procedure to draw the 'Mystic Rose' geometric shape on the screen.
120 Calls a procedure to print the 'Mystic Rose' geometric shape between given vertical limits.
290 Loads X register with number of lines to be printed.
300 YLOOP. Saves X register.
310 Jumps to graphics subroutine.
320 Pulls X register.
330-340 Subtracts four from Y.
350 Decreases X register.
360 Branches to YLOOP if X register is not zero.
370 Returns to BASIC.
380 Graphics. Next character to printer only.
390 Turns graphics mode on for printer.
400 Loads X register (line length +1).
410 Puts X and X1 to zero.
420 A%=0. B%=128. Needles firing = 0.
430 XLOOP.
440 Saves X register.
450 Loads X register with eight (needles).
460 INNER.
470 Saves X register.
480 Point (X1, Y) using OSWORD.
490 Loads accumulator with the result.
500 Branches to zero if point (X1, Y) is equal to zero.
510 Adds B% to A%, adds 1 to needles
520 ZERO. Divides B% by two.
530 Pulls X register.
540-550 Adds four to X.
560 Decreases X register.
570 Branches to INNER if B% 0.
580 Compares needles firing to five.
590 Branches to GO if graphic line is not too heavy.
600 Jumps to DELAY.
610 Next character to printer only.
620 Prints character.
630 Pulls X register.
640-650 Adds 32 to X.
660 A%=0, B%=128. Needles firing =0.
670 Decreases X register.
680 Branches to XLOOP if line is unfinished.
690 Returns from subroutine.
700 DELAY.
710 Saves all registers.
720-730 Sets waiting time.
740-770 Sets interval timers.
790 STAY.
790 Reads interval timer.
800-810 If ,MSB is still negative, branch to STAY.
820 Pulls all registers.
830 Returns from subroutine.

PROCDUMP

160 Assigns Page Zero locations to the vertical screen limits.
170 Assigns a Page Zero location to the vertical length.
180 Calls the machine code print routine.

PROCINIT

220 Dimensions the positions of the vertices.
230 Defines OSWORD and OSWRCH.
240 Dimensions 255 bytes for machine code.
250 Defines where machine code is to start.
260 and 850 Passes DUMP twice for assembly.

PROCDRAW

890 Redefines the graphics origin.
900-920 Calculates and stores the vertices.
930-980 Draws the 'Mystic Rose' geometric shape on the screen.
990 Redefines the origin at (0,0). This is necessary because POINT (X1, Y) is relative to the graphics origin.
1000 Prints the title.

To convert SCREENDUMP for another printer, several changes may be needed.

The eight needles in the Amber printer are horizontal where most others are vertical. In this case, the inside loop of the program would need to scan

CONTINUED OVER

vertically.

The resolution of your printer may be different to the Amber so you must bear in mind that SCREENDUMP is only scanning every fourth screen pixel.

The control code to turn on graphics will vary on other printers, and there may be other differences.

IN CONCLUSION

SCREENDUMP can be used in a graph plotting program in the following way:

LOAD and RUN SCREENDUMP to assemble the machine code.
DELETE lines 10 to 1000 as they are no longer needed.

LOAD the program noting the last block number BB.
LIST the program.
Manually add PROCDUMP to print the chosen part of the screen when required, taking care in placing the graphics origin. It will be necessary to CALL &D00 instead of CALL DUMP in this procedure.

*SAVE "PROGRAM"
0D00 is the start address.

(BB+1)00 is the finish address.
0E00 is the execution address.
This routine aded to an existing program will give hard copy of your graph whenever you need one.

PROGRAM LISTING

```
 10  REM***************
 20  REM*             *
 30  REM* SCREENDUMP *
 40  REM*      BY     *
 50  REM* M. STANGER *
 60  REM*             *
 70  REM***************
 80 :
 90  MODE4:!&FE00=&10200A
100  PROCINIT:REM ASSEMBLES ROUTINE TO PRINT
     LHS OF SCREEN FROM GRAPHICS ORIGIN

110  PROCDRAW
120  PROCDUMP(0,576)
130  END
140 :

150  DEFPROCDUMP(YL,YH)
160  ?&72=YH MOD 256:?&73=YH DIV 256
170  ?&74=(YH-YL+4)DIV 4
180  CALL DUMP
190  ENDPROC
200 :
210  DEFPROCINIT
220  DIM P(10),Q(10)
230  OSWRCH=&FFEE:OSWORD=&FFF1
240  DIM DUMP &FF
250  DUMP=&D00
260  FOR P=0TO 2 STEP2:P%=DUMP
270    [OPT P
290    LDX &74
300    .YLOOP TXA:PHA
310    JSR GRAPHICS
320    PLA:TAX
330    LDA &72:SEC:SBC #4:STA &72
340    LDA &73:SBC #0:STA &73
350    DEX
360    BNE YLOOP
370    RTS
380    .GRAPHICS LDA #1:JSR OSWRCH
390    LDA #&11:JSR OSWRCH
400    LDX #19
410    LDA #0:STA &75:STA &70:STA &76:STA &71
420    STA &80:STA &82:LDA #128:STA &81
430    .XLOOP
440    TXA:PHA
450    LDX #8
460    .INNER
470    TXA:PHA
480    LDX #&70:LDY #0:LDA #9:JSR OSWORD
490    LDA &74
500    BEQ ZERO
510    LDA &80:CLC:ADC &81:STA &80:INC &82
520    .ZERO LSR &81
530    PLA:TAX
540    LDA &70:CLC:ADC #4:STA &70
550    LDA &71:ADC #0:STA &71
560  DEX
570  BNE INNER
580  LDA &82:CMP #5
590  BMI GO
600  JSR DELAY
610  .GO LDA#1:JSR OSWRCH
620  LDA &80:JSR OSWRCH
630  PLA:TAX
640  LDA &75:CLC:ADC #32:STA &75:STA &70
650  LDA &76:ADC #0:STA &76:STA &71
660  LDA #0:STA &80:STA &82:LDA #128:STA &81
670  DEX
680  BNE XLOOP
690  RTS
700  .DELAY
710  PHA:TXA:PHA:TYA:PHA
720  LDA #&FD:STA &8B
730  LDA #&FF:STA &8C:STA &8D:STA &8E:STA &8F
740  LDX #&8B
750  LDY #0
760  LDA #4
770  JSR OSWORD
780  .STAY LDA #3
790  JSR OSWORD
800  LDA &8F
810  BMI STAY
820  PLA:TAY:PLA:TAX:PLA
830  RTS
840 ]
850  NEXT
860  ENDPROC
870 :
880  DEFPROCDRAW
890  VDU 29,288;288;
900  FOR A=0 TO 10
910    P(A)=200*SINRAD(A*36):Q(A)=200*COSRAD(A*36)
920    NEXT
930  FOR I=1 TO 10
940    MOVEP(I),Q(I)
950    FOR J=1 TO 10
960      DRAW P(J),Q(J)
970      MOVE P(I),Q(I)
980      NEXT:NEXT
990  VDU 29,0;0;
1000 PRINTTAB(4,30);"Mystic Rose"
1010 ENDPROC
```

THE NEW AND UNIQUE C.A.D. PROGRAMME FOR THE BBC (32K)

AT A VERY AFFORDABLE PRICE

Ideal for teachers, designers, artists, technical drawing and numerous other applications including your own form and stationery design etc.
This programme must be seen to be appreciated – your imagination is the only factor to limit its individual applications.

- Modes 0,1,2,4,5 (can be changed when programme is running).
- Multiple display of arrays enables infinite complexity.
- **FUNCTIONS**: Line, rectangle, triangle, circle, text (upper and lower case) and colour pallet (8 colours and flashing).
- **DRAWING AIDS**: Alignment grid, circle copy, delete, free memory, purge memory, variable cursor speed, clear screen and redraw.
- Shapes can be filled or outlined (no need for Fill Routines).
- Save and load to tape in about *20 seconds*.
- SPECIAL FACILITY – Rubber band mode – A very flexible and variable line drawing facility – must be seen.
- Free "redraw" routine to enable the pictures created to be displayed in your own programmes.

Grafstik (for joystick) **Grafkey** (for keyboard)
Grafdisk (combines Grafstik and Grafkey)
Fantastic Value only £7.95 inc. p&p
Grafdisk £12.95

JOYSTICK
A pair of fully proportional joysticks. Compact, handy sized. Smart injection moulded cases **£17.95**
CASSETTE LEADS With Motor Control
DIN to DIN & Remote (BBC or ORIC) **£2.50**
DIN to 3 jacks (BBC or ORIC) **£2.50**
Dragon Lead (DIN to DIN Or DIN to 3 jacks) **£2.50**
TEXAS T1 99/4A (Single Recorder) **£4.95**
MONITOR LEADS
BNC to Phono **£2.95**
Phono to Phono **£1.50**
6 pin DIN to DIN RGB LEAD **£2.50**
SERIAL PRINTER LEAD (Acorn Specs) **£7.50**
All Leads are Top Quality and carry an Unconditional Guarantee.

Access

CLARES MICRO SUPPLIES
222 TOWNFIELD ROAD,
WINSFORD, CHESHIRE, CW7 4AX
TEL: 060 6551374

BBC Micro-Aid

SOFTWARE - Programs that are guaranteed to run! Save hours of work and worry with these utilities and practical programs on cassette or disc.

102	**CASHBOOK**	Double entry 4 columns with accounts	£ 5.95	B
103	**LEDGER**	Complements CASHBOOK with ageing & analysis	£ 5.95	B
104	**MAILING**	Holds 218 addresses. System with 6 options, 2 Sorts, Labels, 2 Searches & Update	£ 5.95	B
105	**PAYROLL (W or M)**	In two parts to handle weekly or monthly PAYE & NI for 100 employees. Supported.	£13.90	B
105a	**Manual**	Extra (No VAT)	£ 2.50	B
106	**MEMO-CALC**	Database/Calcsheet with up to 255 columns, string or numberic data, sorts, searches, calculations, with full print facility.	£ 9.95	B
106a	**Manual**	Extra (No VAT)	£ 2.00	
201	**CARDS**	Beat Bruce Forsyth at his own game.	£ 2.95	A/B
202	**BATTLE**	Fast moving game simulation of a minefield.	£ 2.95	B
203	**HANGMAN**	Word game. French, German, Italian, Spanish.	£ 7.95	B
301	**BANNER**	Paper printout of large text & graphics.	£ 2.95	A/B
302	**DISTANCES**	Graphic maps of U.K., EUROPE & WORLD. Calculate distance between any two places.	£ 3.95	B
303	**FLAGS**	Full colour flags of the world & questions.	£ 3.95	B
304	**STATPAK**	Statistics offering over 30 results.	£ 8.95	B
501	**SEARCHBAS**	PROC to search a BASIC program & alter it.	£ 1.95	A/B
502	**PROCVAR**	PROC to list variables used in a program.	£ 1.95	A/B
503	**PROCFLUSH**	PROC to clear Resident Integer Variables.	£ 1.00	A/B
504	**PROCAID**	A combination of programs 501, 502 & 503.	£ 3.45	A/B
505	**DEFCHR**	Design, display & store graphic characters.	£ 2.95	A/B
506	**SORTM/C**	Machine code sort for up to 255 integers.	£ 1.00	A/B
507	**SORBAS**	A very fast BASIC sort. 1000 items in 42s.	£ 1.00	A/B
508	**UTILITY-A**	Combination of 501 to 507. Superb value.	£ 5.95	A/B
600	**FORTH**	Second language ROM for either OS.	£34.74	B
601	**WORDWISE**	ROM Superb fast & easy wordprocessor OS 1.0	£34.74	B
701	**BOOKS**	Various titles for the BBC Micro from (No VAT)	£ 6.95	
801	**CASSETTES**	C12 Computer quality boxed in 10's.	£ 4.50	
810	**DISCS**	MEMOREX Soft Sectored 40 track 5.25"	£19.95	
901	**NEW EPSON**	FX-80 F/T, 160CPS, 3 FOUNTS, Graphics.	£379.00	
905	**SEIKOSHA**	GP-100a Printer 50cps, 80 columns, tractor	£195.00	
910	**NEW TEAC**	Slimine Disc drives suitable for BBC Micro with Power supply. Format disc & Manual.		
	100K	Single Sided, 40 Track. OTHERS AVAILABLE.	£199.00	
920	**VDU STAND**	Stainless steel support protects your micro	£19.95	

ADD VAT TO ALL PRICES. FOR COPY ON DISC ADD £1.00.
NO PACKING CHARGE.

If you want further information before parting with your hard earned cash send for our new brochure to:-

Micro-Aid (BC)
25 Fore Street, Praze, Camborne, Cornwall TR14 0JX.
Tel: 0209-831274

TV CHOICE

Electronequip

(Authorised BBC Dealer, and service centre)

NEW PRODUCTS

Utility Disc for BBC Contains VER FORM 35 FORM40 and FORM80 ..... **£9.95**

BBC Sparkjet Printer New quiet printer for BBC. Friction and tractor feed 80cps. **£379.50**

Torch Z80 Disc pack 800K dual disc drive plus Z80 processor with CMP compatible operating system **£897.00**

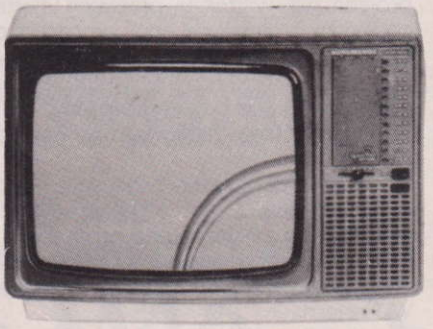
TORCH Computer 800K to 21.4M disc drives. High res. colour monitor. Plus auto-dial modem..... **From £2795.00 + VAT**

New Epson FX80 FX80 160cps printer in stock. Friction and tractor feed + proportional spacing..... **£458.85**

BBC 2	BBC Model B Micro Computer	£399.00
BBC 3	BBC Model A Micro with 32K	£333.50
BBC 4	BBC Model A Micro with 32K and VIA	£339.50
BBC 5	BBC Model B with Disk Interface	£469.00
BBC 6	BBC Model A with Econet Interface	£356.00
BBC 7	BBC Model B with Econet Interface	£456.00
BBC 8	BBC Model B with Disc & Econet Interface	£526.00
BBC 21	BBC Model A to B Upgrade	£99.82
BBC 28	Econet Upgrade for BBC	£92.00
BBC 23	BBC Acorn Memory Upgrade for Model A	£34.50
BBC 27	Disc Upgrade for BBC B (inc fitting)	£92.00
BBC 30	BBC 14" Colour Monitor	£287.50
BBC 33	Sanyo SM12N Green Monitor 15MHz	£90.85
BBC 34	BMC 12E Green Monitor 18MHz	£113.85
BBC 35	Karga K12A 12" Orange Monitor	£129.95
BBC 41	BBC Single 100K 5.25" Disc Drive (AND01)	£265.00
BBC 43	BBC Dual 800K 5.25 Disc Drive (AND02)	£803.85
BBC 44	Single Disc Drive (100K) for BBC (Teac)	£205.85
BBC 45	Single Disc Drive (200K) for BBC (Teac)	£263.35
BBC 46	Single Disc Drive (400K) for BBC (Teac)	£343.85
BBC 47	Dual Disc Drive (200K) for BBC (Teac)	£411.70
BBC 48	Dual Disc Drive (400K) for BBC (Teac)	£526.70
BBC 49	Dual Disc Drive (800K) for BBC (Teac)	£687.70
BBC 50	Epson FX80T 160cps Printer	£458.85
ATM 2	Acorn Atom assembled 12K ram	£184.00
ATM 26	Atom New Power Supply 1.8A	£9.66
ATM 21	Floating Point ROM for Atom	£21.85

Large stocks of software for BBC and Atom, Business, Games and Educational. Send for comprehensive lists.

Large stocks. Prices inclusive of VAT. All prices inclusive of postage except micros. £3.00. All Upgrades etc. are fitted free of charge and the computer fully re-tested. Access and Barclaycard Welcome.



14" Colour portable Monitor/TV
This Monitor/TV us not a modified television as many TV/Monitors are, but a 14" Monitor/TV which has been designed to perform both functions. It has RGB and Composite video and sound. An RGB cable for a BBC is supplied as standard.
£259.90 Trade enquiries welcome

ACORN COMPUTER

Electronequip BBC

36-38 West Street, Fareham, Hants TO16 0GN (0329) 230670

GEMIN

PERFORMAN

It can do a powerful job for yo

SPECIAL LIMITED OFFER

Buy just any two programs at £19.95 and take one at £19.95 FREE!

Here's a range of software for the independent businessman that's designed to harness the power of your micro to deliver the vital information you need in all key areas of your business. A breakthrough on both price and performance, each program is fully tested and comes with all the documentation back up you need.

"Gemini's range of software is in the vanguard of the releases for 'serious' micro users . . ."
(Which Micro and Software Review)

CASH BOOK ACCOUNTS PROGRAM FOR BBC MICRO . . .£95.00

New

One of the most innovative business programs on the market. Most serious accountancy packages are written and coded by professional and competent programmers. The Gemini Cashbook Accounting program was written by practising Chartered Accountants and coded by professional and competent programmers. This is a fundamental difference.

This practical program is simple to use and will replace your manual cash and bank records and by giving you instant management information, it may even put your accountant out of job!

With exceptionally exhaustive user documentation, full technical back up and product update policy this program will increase the efficiency and profitability of your business. Take a look at the information this program will provide.

* summary of VAT information to enable you to complete your VAT returns
* cumulative receipts and payments report analysed over the standard profit and loss and balance sheet heading.
* option for departmental analysis of sales and purchases
* print out of all transactions
* journal routine for entering transfers between accounts and year end adjustment for debtors, creditors etc.
* year end trial balance
* profit and loss account and balance sheet.

These statements can be produced at what ever interval you require e.g. monthly, quarterly or annually.

Coming soon:– Integrated Sales + Purchase Ledgers

". . . the systems worked immaculately when tested . . ."
"Mailist is a very professional piece of software . . ."
(Which Micro & Software Review Feb 83)

SPREADSHEET ANALYSIS
BEEBCALC £19.95
DRAGONCALC £19.95

New

FOR BBC AND DRAGON 32. Spreadsheet processors have proved to be important tools for using micros in business, scientific and domestic financial applications.

Without any programming knowledge at all, you may:–

* Set up a computerised spreadsheet, with chosen row and column names.
* Specify formulae relating any row or column to any other.
* Enter your source data and have the results calculated.
* Save the results on tape (or disk – BBC) for later reloading and manipulation.
* Print the tabulated results in an elegant report format.
* Experienced users may access saved files and write their own reporting or graphics presentation programs for the results.

Some typical applications:–

* Small business accounting applications, e.g. profit and loss statements and cashflow projections, break-even analyses etc.
* Investment project appraisal – anything from double glazing to oil rigs!
* Comparing rent/lease/buy options
* Processing the results of scientific experiments or field studies
* Engineering calculation models
* In fact, anything that involves repeated re-calculation of results presented in tabular or spreadsheet format.

Program Availability Chart:–

	Database	Stock Control	Mailist	Invoices & Statements	Spread sheet Analysis	Cashbook Accounting	Word processor	Home Accounts	Commercial Accounts
Sinclair Spectrum 16k or 48k	●	●	●					●	●
Dragon 32k or 64k	●				●			●	
VIC20 (16k+)	●	●	●	●				●	●
Sinclair ZX81 (16k+)	●								
Grundy Newbrain	●								
Texas TI99 4A	●								
Osborne 1 Atari 400/800	●								
Sharp MZ80A	●	●	●	●				●	●
Sharp MZ80K	●	●	●	●				●	●
Sharp MZ80B	●	●	●	●				●	●
BBC micro model A or B 32K	●	●	●	●	●	●	●	●	●

'S NEW
CE SOFTWARE
r business at petty cash prices.

INVOICES AND STATEMENTS . . . £19.95
Compatible with most micros. See table. Ideal for the small business. A complete suite of programs together with generated customer file for producing crisp and efficient business invoices and monthly statements on your line printer. All calculations include VAT automatically, and the program allows your own messages on the form produced. This program gives you superb presentation and saves time on one of the most tedious tasks in the office.

COMMERCIAL ACCOUNTS . . . £19.95
Compatible with most micros. See table. A gem of a program, all for cassette, with the following features:– Daily Journal. Credit Sales. Cash Sales. Credit Purchases. Purchases – other. Sales Ledger. Purchase Ledger. Bank Account. Year to date summary. A fully interactive program suitable for all businesses. Files can be saved and loaded and totals from one file carried forward to another on cassette. Particularly useful from a cash flow point of view, with an immediate accessibility to totals for debtors and creditors. Bank totally supported with entries for cheque numbers, credits and, of course, running balance.

MAILING LIST . . . £19.95
Compatible with most micros. See table. A superb dedicated database to allow for manipulations of names and addresses and other data. Gemini's unique 'searchkey' system gives you a further ten 'user-defined parameters' to make your own selections. Features include the facility to find a name or detail when only part of the detail is known, it will print labels in a variety of user specified formats.

DATABASE . . . £19.95
Compatible with most micros. See table. The program that everyone needs, the most valuable and versatile in your collection. Facilities include sort search, list print if required. Can be used in place of any card index application; once purchased you can write your own dedicated database to suit your particular needs with a limitless number of entries on separate cassettes.

STOCK CONTROL . . . £19.95
Compatible with most micros. See table. Dedicated software with all that's necessary to keep control of stock. This program will take the tedium out of stock control and save time and money. Routines include stock set up, user reference number, minimum stock level, financial summary, line print records, quick stock summary, add stock, delete/change record and more.

HOME ACCOUNTS . . . £19.95
Compatible with most micros. See table. Runs a complete home finance package for you with every facility necessary for keeping a track of regular and other expenses, bank account mortgage, H.P. etc. This program also allows you to plot graphically by Listograms your monthly outgoings.

WORD PROCESSOR . . . £19.95
Compatible with most micros. See table. This program features routines found in much larger and more expensive packages with a typical word length of 5-6 letters it allows for around 1000 words in memory at one time. Ideal for the user who requires a simple program to write letters on his computer. Features include, block delete, block insert, search and replace, edit text, display text and more.

Dealer/Trade enquiries invited – generous trade discounts for quantity

Special ACCESS card instant sales hotline for GUARANTEED PROMPT despatch . . .

Tel: 03952-5165

24 hr Ansaphone Service.

All enquiries other than credit card sales to 03952-5832

Gemini. Functional Software Specialists. 9, Salterton Road, Exmouth, Devon.

Tick the box for Program you require. Prices include V.A.T. and Package and Postage.
Please supply the following cassette software.

Database	£19.95 ☐	ZX81 16K Database	£9.95 ☐
Stock Control	£19.95 ☐	BBC Cash Book disk or tape	£95.00 ☐
Mailing List	£19.95 ☐	BBC Disks – other titles	£23.95 ☐
Invoices and Statements	£19.95 ☐	Osborne Disk Database	£23.95 ☐
Commercial Accounts	£19.95 ☐	Word processor	£19.95 ☐
Home Accounts	£19.95 ☐	Beebcalc	£19.95 ☐
		Dragoncalc	£19.95 ☐

Name_____

Address_____

Machine Type_____ Memory Size_____

I enclose_____

Make cheques and postal orders payable to Gemini Marketing Ltd.

Diners Card Number_____ Access Number_____

Signature_____

Gemini. Functional Software Specialists, 9 Salterton Road, Exmouth, Devon.
EX8 2QG

ABC 1

Labelling on the Screen

J Ruston

Place labels wherever you want on your BBC's screen. We lift the restrictions for you.

Text on the BBC computer is normally made up of 64 dots arranged in a matrix of 8 by 8. These dots are selectively turned on and off to create the different letters. There is another, more versatile, way of doing it, however.

Graph plotters, such as the printer/plotter which mates with the Sharp PC-1500, make up the letters out of little lines, in much the same way as we draw them.

THE MICRO MIMIC

This program mimics a plotter in producing text on the screen of the BBC Micro. It is made up a string which has to be initialized in line 10, a procedure stretching from line 10000 to line 10850, and a small demo program from line 20 to line 90.

The procedure PROCword is called with the following parameters:

—A string of text to be printed. This string can be made up of any characters in the first 64 ASCII characters. It is easy to extend the program to print lower case as well.
—The co-ordinates from where the text should be drawn.
—The width of each letter.
—The height of each letter.
—The rotation with which the text should be drawn. This should be between -180 and 180 degrees. A rotation of zero draws the text along the screen as normal, and a rotation of -90° draws the text up the screen, suitable for labelling the Y axis of graphs.
—The slope of the text. Zero will draw upright text, a positive number will draw text which slopes to the right, and a negative number, text which slopes to the left.

Thus, you have control over every aspect of the text. It is worth pointing out that unless you have a monitor with Mode 0, the text has to be drawn fairly large to be readable.

I have replaced the hash symbol with the pound sign in the character set of the program, because it is more useful.

If you have a model A, you can run the program in Mode 4, by removing the GCOL statements, and the VDU 19 statement.

PROGRAM DESCRIPTION

Having decided to write the program, the most difficult thing to do was to work out the best way of storing lines that go to make up each character.

I first drew each character on 64 characters could be adequately represented using a five by seven matrix of 'nodes' which could be selectively joined together. So, I drew out this matrix, and labelled each 'node' like this:

6	D	K	R	Y
5	C	J	Q	X
4	B	I	P	W
3	A	H	O	V
2	9	G	N	U
1	8	F	M	T
0	7	E	L	S

For example, A could be drawn by moving to 0, drawing to 4, drawing to K, drawing to W, drawing to S, and then drawing to crossbar from 2 to U.

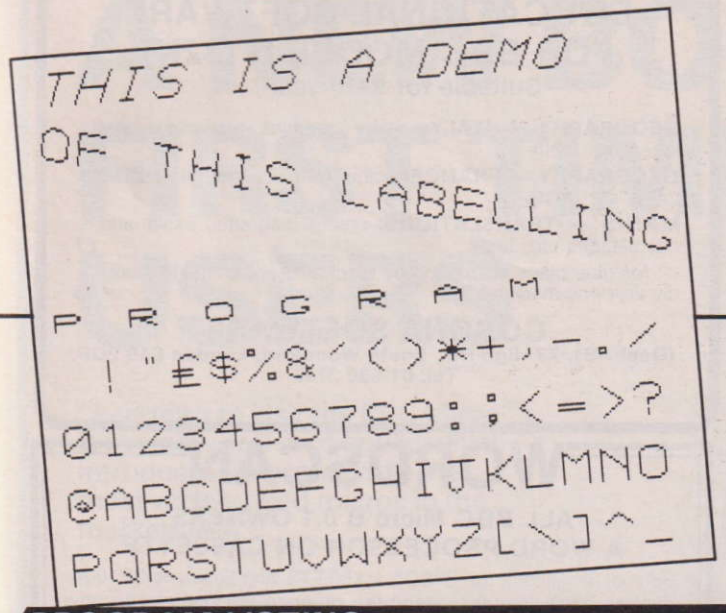
The code chosen for encoding each character was that each would be represented by a string of characters from the matrix, and each node mentioned in the string would be joined to the next one. Thus, for the letter L, we would have S06. An extension to allow for the non-contiguous characters, such as ':' was that if a letter was preceded by Z, it was to be taken as a MOVE, and not a DRAW. I assumed that every letter's data would start off with Z, so the final code for the exclamation mark was 'KGZFE', which is almost more compact than the eight character generator bytes required in the computer's ROM for that character.

LABOUR INTENSIVE

The next stage was to laboriously work out the DATA for each character, which you can see in the program listing. All the characters have turned out to be pleasing, with the exception of

LINE DESCRIPTION

10020	H% gives the overall width of the entire letter; so to get the distance between two horizontal nodes, it has to be divided by five, since there are five nodes across each character.
10030	And the same goes for the height of each character.
10040	Later on, during the section of the program which deals with the rotation of the text, the cosine and sine of the rotation are required, so they are worked out once only.
10050	See line 10040.
10060	Starts a loop through all the letters to be printed.
10070	RESTOREs the data pointer to the DATA for the next character to be printed. If you RENUMBER the program, you will have to alter the 10220 to be the new line number of the first DATA statement.
10080	Reads the DATA for the character.
10090	Adds a Z to the start of it, because it is assumed that each string starts with the code for a move.
10100	Starts a loop through all the characters in the DATA string.
10110	Picks out the relevant character.
10120	Works out the position of the letter in the string L$, then decrements it by one.
10130	Works out the Y co-ordinate of the node, and multiplies it by the height.
10140	Does the same for the X co-ordinate, and adds the slope variable.
10150	If the character is not a Z, then draws to the point. The peculiar arguments to the PLOT statement cope with the rotation. D% is four when a move is to be made, and five when a draw is to be made.
10160	Sets D% to a DRAW, its default.
10170	If the character was a Z, then sets D% to MOVE.
10180	Ends the loop through each datum.
10190	Moves the X and Y co-ordinates along, so the letters are not drawn on top of each other. Again, the movement is complicated by the rotation feature.
10200	Ends the loop through all the characters.
10210	Ends the whole procedure.



the '@' symbol, which I just could not get to look right.

The next stage was to find some way of interpreting this DATA to get the letters on the screen. This is where the string in line 10 comes in. All that needs to be done is work out the position in the string of each letter in the data string. If this number were decremented by one, you have a unique number between 0 and 34 for each of the 'nodes'. The X co-ordinate of each node is given by this number DIV 7, and the Y by the number MOD 7.

Having that background knowledge, it is safe to examine each line of the procedure individually.

PROGRAM LISTING

```
   10 L$="0123456789ABCDEFGHIJKLMNOPQRST
UVWXYZ"
   20 MODE 1:VDU 19,3,4;0;
   25 GCOL 0,1
   30 PROCword("THIS IS A DEMO",0,800,60
,60,0,0.5)
   35 GCOL 0,2
   40 PROCword("OF THIS LABELLING",0,720
,60,60,10,0)
   45 GCOL 0,3
   50 PROCword("P R O G R A M",0,450,60,
30,0,0)
   55 GCOL 0,1
   60 PROCword(" !""#$%&'()*+,-./",0,350
,60,60,0,0)
   65 GCOL 0,2
   70 PROCword("0123456789:;<=>?",0,250,
60,60,0,0)
   75 GCOL 0,3
   80 PROCword("@ABCDEFGHIJKLMNO",0,150,
60,60,0,0)
   85 GCOL 0,1
   90 PROCword("PQRSTUVWXYZ[\]^_",0,50,6
0,60,0,0)
 9999 END
10000 DEF PROCword(S$,X%,Y%,H%,V%,R%,S)
10010 LOCAL A$,C$,T%,L%,M%,X,Y,G%
10020 H%=H% DIV 5
10030 V%=V% DIV 7
10040 X=SIN(RAD(R%))
10050 Y=COS(RAD(R%))
10060 FOR G%=1 TO LEN(S$)
10070 RESTORE (10220+(ASC(MID$(S$,G%,1))
-32)*10)
10080 READ A$
10090 A$="Z"+A$
10100 FOR T%=1 TO LEN(A$)
10110 C$=MID$(A$,T%,1)
10120 I%=INSTR(L$,C$)-1
10130 M%=(I% MOD 7)*V%
10140 L%=(I% DIV 7)*H%+M%*S
10150 IF C$<>"Z" THEN PLOT D%,X%+L%*Y+M%
*X,Y%-L%*X+M%*Y
10160 D%=5
10170 IF C$="Z" THEN D%=4
10180 NEXT T%
10190 X%=X%+6*Y*H%:Y%=Y%-6*X*H%
10200 NEXT G%
10210 ENDPROC
10220 DATA 4
10230 DATA KGZFE
10240 DATA KJZRQ
10250 DATA XRKC70SZ30Z2N
10260 DATA WQC4AOUM82ZEK
10270 DATA 0XZ6DC56ZSTMLS
10280 DATA SMNHPQKD54AHA217EU
10290 DATA BR
10300 DATA RB9L
10310 DATA DPN7
10320 DATA 1XZ5TZJFZ3V
10330 DATA 3VZJF
10340 DATA GF7
10350 DATA 3V
10360 DATA 8FE78
10370 DATA X1
10380 DATA 15DRXTL71X
10390 DATA CKE7L
10400 DATA 5DRX0S
10410 DATA 5DRXW0A0UTL71
10420 DATA LR3V
10430 DATA Y630UTL71
10440 DATA XRD517LTU03
10450 DATA 6YXHE
10460 DATA 5DRXW0A217LTU0A45
10470 DATA 17LTXRD54AV
10480 DATA 8FE78ZCJIBC
10490 DATA 7FG98FZDKJCD
10500 DATA R3L
10510 DATA 3VZ2U
10520 DATA DV7
10530 DATA 5DRXWOHGZFE
10540 DATA TL715DRXVNGAIOG
10550 DATA 04KWSZ3V
10560 DATA 06RXW030UTL0
10570 DATA TL715DRX
10580 DATA 06RXTL0
10590 DATA S06YZ3V
10600 DATA 06YZ3V
10610 DATA NUTL715DRX
10620 DATA 063VYS
10630 DATA 7LEKDR
10640 DATA 217FK6Y
10650 DATA 063AYAS
10660 DATA 60S
10670 DATA 06IYS
10680 DATA 06UYS
10690 DATA 15DRXTL71
10700 DATA 06RXW03
10710 DATA 15DRXTL71ZGS
10720 DATA 06RXW03AS
10730 DATA XRD54A0UTL71
10740 DATA 6YKE
10750 DATA 617LTY
10760 DATA 62EUY
10770 DATA 60GSY
10780 DATA 65TSZ01XY
10790 DATA 65HEHXY
10800 DATA 6YX10S
10810 DATA RD7L
10820 DATA 5T
10830 DATA DRL7
10840 DATA 4KW
10850 DATA 0S
```

BBC MICRO INSTANT MACHINE CODE!

Yes, it's true. Instant machine code from a good subset of BBC BASIC. Type your BASIC program into your model B BBC Micro, trigger the compiler, and your program is changed almost instantaneously into superfast machine code.
For £34.95 you get: Cassette version of the complete compiler (along with a version of the compiler for use with discs, ready for when you upgrade, the disc version being dubbed on the cassette after the cassette version); complete compiler listing; extensive documentation and instructions. The compiler was written by Jeremy Ruston.

THE BBC MICRO REVEALED

By Jeremy Ruston

'. . .destined to become the bible of all BBC microcomputer users. . .' (Personal Computing Today). If you've mastered the manual, then this book is for you. Just £7.95

LET YOUR BBC MICRO TEACH YOU TO PROGRAM

By Tim Hartnell

'. . .takes you further into the cloudy areas of the BBC machine than anything else I've yet seen. . .' (Computer and Video Games). If you're just starting out in the world of programming, then this book is the one for you. Forty complete programs, including Othello/Reversi, Piano and a host of dramatic graphic demos. Just £6.45

Interface, Dept. BBC

44-46 Earls Court Road, London W8 6EJ

Please send me:

( ) INSTANT BBC MACHINE CODE–tape and book–£34.95

( ) THE BBC MICRO REVEALED–Ruston–£7.95

( ) LET YOUR BBC MICRO TEACH YOU TO PROGRAM–Hartnell–£6.45

I enclose £

Name .....

Address .....

EDUCATIONAL SOFTWARE FOR BBC MODEL B (32K)

Suitable for 9–16 year olds

GEOGRAPHY — ITALY: colour graphics, mountains, rivers, cities, etc., tests **£7**

GEOGRAPHY — FRANCE: colour graphics, mountains, rivers, cities, etc., tests **£7**

MATHS — TRANSLATIONS: colour diagrams, explanations, calculations etc., tests **£7**

for use: class instruction by teacher or individual learning dis versions available **£9**

CORONA SOFTWARE
(Dept AB), 73 High Rd., South Woodford, London E18 2QP
Tel: 01-530 3747

WORDSCAN

ALL BBC Micro B 0.1 OWNERS
A WORD PROCESSOR ON CASSETTE

Special facilities of the EPSON MX80 FT3, and Seikosha GP100A printers easily requested and the program will operate on any "machine compatible" printer to produce over 400 by 40 character print lines.

Bureau Service available for those without a printer.
"Edit" your text, "Save" it on cassette, "Post" it to me for printing using the EPSON MX80 FT3 printer.

For Demonstration cassette send £2 deposit + £1 p&p.
Otherwise, full details of the Package, priced at **£15**, and the Bureau Service (minimum charge £2 or 2000 words) can be obtained from:

DIAL SOFTWARE
72 Downend Road, Downend, Bristol BS16 5UE.

FRENCH ON THE BEEB!

Educational Software for the BBC Model B

TEST yourself, **BROWSE** at random, use as a **REFERENCE**

All tenses of over 150 French **IRREGULAR** verba. For all ability levels cassette **£19.30**

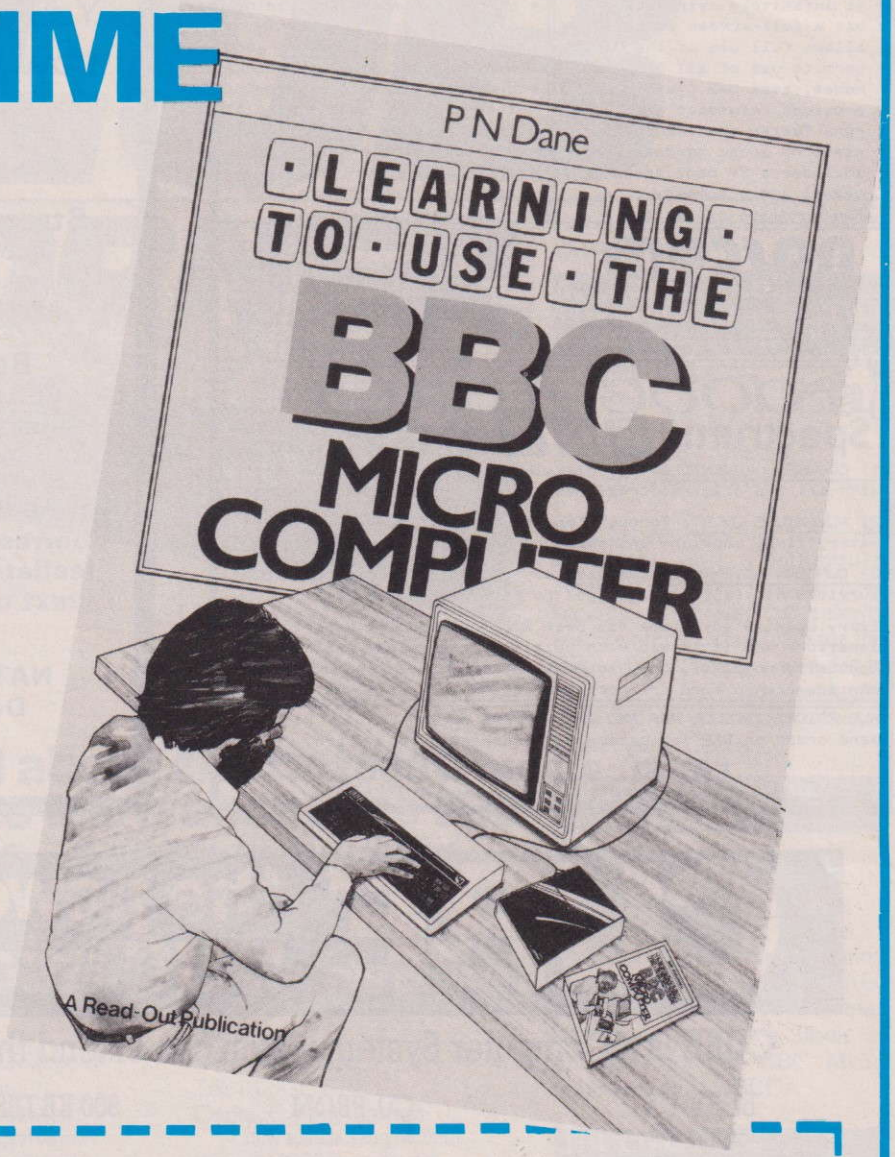
A unique practical programme to challenge beginners. O & A level students and even French graduates

* menu driven
* english headings
* 3 levels
* attractive colour format
* test with time limit
* choice of tenses
* fully accented
* vetted by native speakers

RUSSIAN VERBS available, s.a.e. for details.

CARSENDALE ENTERPRISES
Dept B, 44 Kingsway, Stoke on Trent
Staffs ST4 1JH

A NEW BOOK FOR FIRST TIME USERS



"Learning to Use the BBC Micro" is in a new series of books which introduces newcomers to the most widely used micros in the marketplace.

The books assume absolutely no knowledge about computers and the reader is shown even the most fundamental operations such as "switching on" and "loading a program". The books lead the reader through simple programming and then on to graphics, with several programs which show how to achieve pictures and even animation!

The user-friendly approach is consistent throughout the text – not only are program listings clearly shown, but in many cases, a photograph is included to show what the program looks like when actually loaded and run!

All books in the series are £5.95 (incl. postage).

Gower

"The Learning to Use" series will bring comfort and help to many first time microcomputer users. The books represent the first genuine approach to assisting the beginner to get to grips with the best selling personal microcomputers . . ."

Communicate

Order Form to: **READ-OUT BOOKS AND SOFTWARE**

8 CAMP ROAD, FARNBOROUGH, HANTS, GU24 6EW.
24 Hour Answering Service. Telephone 0252 510331/2

Name _____

Address _____

_____

Please send me .....copy/ies of

☐ Learning to Use the BBC Microcomputer

@ £5.95 including postage and packing. (Allow 14 days for delivery.)

Make cheques payable to Read-Out Publishing Company Ltd.

I enclose my cheque for £.....

Please debit my Access ☐ Number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Signed _____ Date _____

BBCCOM5

CLUB CORNER

Find about the hottest spots to learn about the BBC Micro. Are there any in your area?

It's easy to feel you are alone when you are sitting in front of your micro wondering why it won't do what you think you've instructed it to do! But you need never be alone again. User groups are springing up all over the place and can be an invaluable source of help, enjoyment and inspiration.

We list here some of the clubs that are particularly interested in the BBC Micro. If you don't see one in your area, why not start one up and let us have details? Please remember though, that this list is by no means comprehensive.

BEEBUG

The National Independent User Group for the BBC Micro
33 St. Julians Road,
St. Albans,
Hertfordshire.
Contact: D E Graham or Sheridan Williams

BEEBUG runs a regular newsletter (10 issues a year) including program listings, hardware and software tips, reviews and advice, all exclusively devoted to the BBC Micro. Membership is available for £4.50 for six months or £8.50 for the full year's subscription.

LASERBUG

10 Dawley Ride,
Colnbrook,
Slough,
Berkshire SL3 0QH.
Contact: Paul Barbour

LASERBUG started off life as a BBC User group based around London, but it is now an international group with members in 14 countries worldwide. A comprehensive magazine is sent to all members monthly. Local meetings are arranged via the newsletter. Annual membership is £12.00 which includes 12 copies of the magazine plus special members only discounts.

NORTHWICH & DISTRICT BBC MICRO USER GROUP

Room B12a,
Northwich City College of Higher Education,
Ipswich Road, Norwich,
Norfolk NR2 2LJ.
Tel: 0603-60011 ext 233
Contact: Paul Beverley

A local support group, membership will cost you £2.00 for the rest of the year unless you are a student or OAP in which case it will cost you £1.00.

COMPUTERTOWN UK!

7 Collins Drive,
Eastcote, Middlesex HA4 9EL.

A nationwide network of computer literacy centres. The idea started in the USA and was brought across and championed by David Tebbutt, the then Editor of *PCW*. Many of the local centres are doing sterling work for the BBC Project and, as the whole idea of CTUK! is to provide free access, they are well worth checking out.

COMPUTER USERS CLUB

72 Sidmouth Road,
Welling,
Kent DA16 1DS.
Tel: 01-304 3910
Contact: Tony Latham

The club produces a monthly printout of software ideas for the BBC Micro, programs and advice on programming technique.

CATERHAM LEISURE CENTRE COMPUTER CLUB

Caterham Leisure Centre,
Godstone Road,
Caterham,
Surrey CR3 6RE.
Tel: Caterham 48304 (M Goldsbrough) or Caterham 43316 (J Hodges)
Contact: M Goldsbrough (Centre Manager) or J Hodges

The club has started at the Leisure Centre which has a Model B BBC Micro available. Meetings are on Thursday nights at 8.00 pm and new members (and their micros) are welcome to come along.

THE FAREHAM AND PORTSMOUTH AMATEUR COMPUTER CLUB

22 Sandy Close,
Petersfield,
Hampshire GU31 4HF.
Tel: 0730-4059 (evenings)
Contact: Peter Smith

Established back in 1980, the Fareham and Portsmouth Amateur Computer Club have recently organised a referral service and a Users Club for the BBC Micro. The group meet at 7.00 pm on the third Monday of each month at the Portchester Community Centre.

INDEPENDENT BBC USERS' GROUP

Dept. 1A,
44-46 Earls Court Road,
London W8 6EJ.

This group has been in operation since October 1981, and is run by Jeremy Ruston and Tim Hartnell who have written a book of programs for the BBC Micro published by the BBC.

THE COMPUTER REFERRAL SERVICE

PO Box 7,
London W3 6XJ

This acts as a central clearing house for information about the BBC Computer Literacy Project. Please remember to enclose a large SAE and, most important, to write on your envelope just what information you require: User Groups, Software, General Query, etc. They have also published a large number of factsheets about various aspects of the project: regional groups, suggested books, micros and the small business, micros and education, and jobs in computing. These are available free on receipt of a large SAE—again please mark it clearly.

Graphics Printing

N Fox

You can create patterns in all shapes and sizes on your computer's screen. Read on to find out how to keep them forever in printout form.

The BBC Micro is an excellent machine for graphics. Not only is this good for games, it can also be used to produce splendid graphs, charts and figures on the screen.

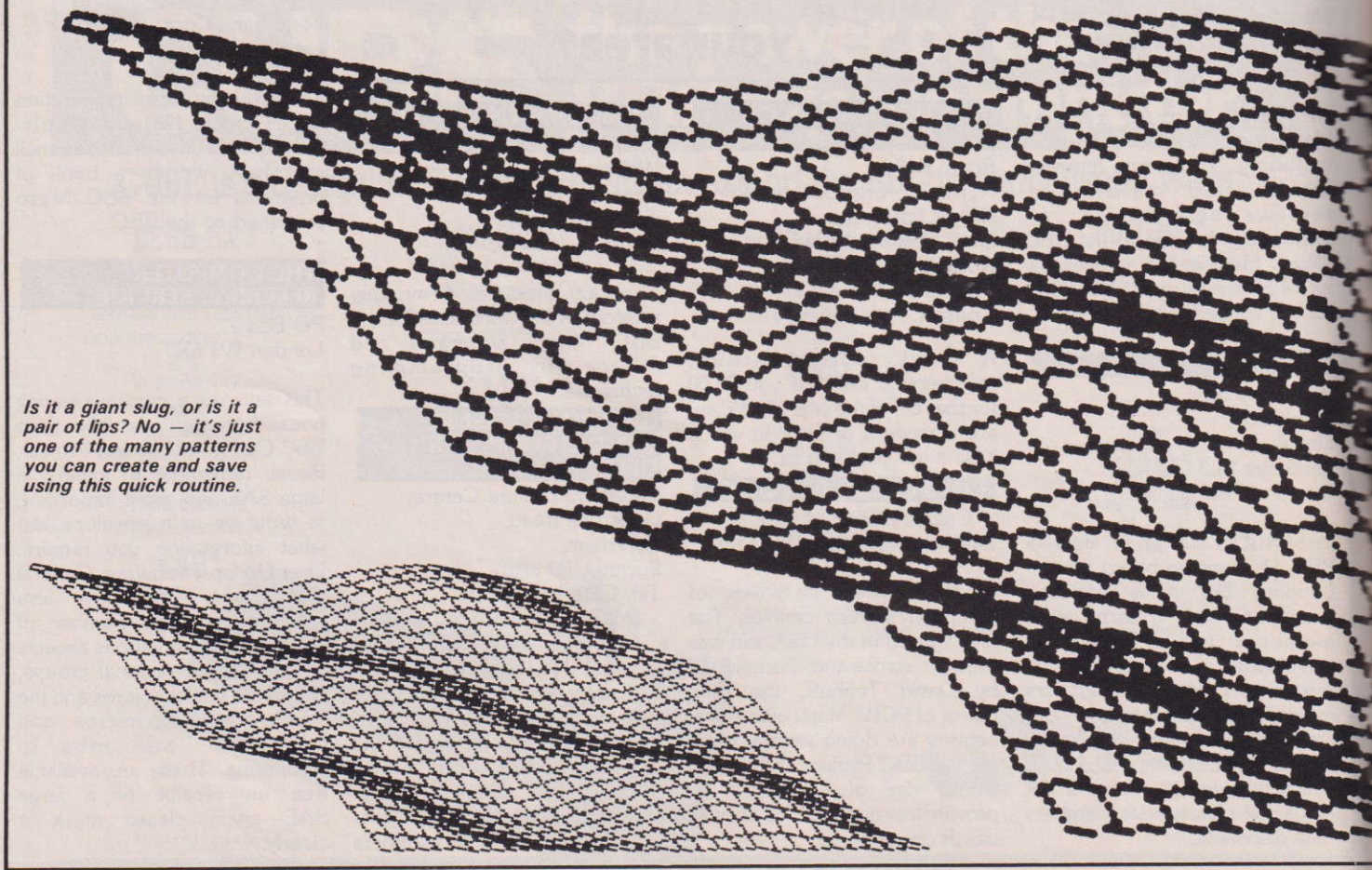
PRINTING ON AN EPSON

When I got to know mine better, I thought how nice it would be to have the facility for putting them onto by Epson printer, so I wrote a program in BASIC to do this. It worked . . . but how slowly! One screenful took about 15 minutes to print. So, why not do it in machine code? The BBC machine has an assembler which can be entered from a BASIC program, which makes it very simple to use. The result is shown in the listing but the development of the program led to some interesting discoveries about the computer.

A CLOSE ENCOUNTER

The first problem you encounter is the way the screen memory works. As you can see from Fig 1, the first character position on the screen (in MODE 0) uses eight bytes of screen memory. Each bit in each byte corresponds to one pixel or one blob of light on the screen. When using the other modes, these bytes also contain colour information which makes things much more complicated. And in any case, the best resolution is from MODE 0, so we'll stick to that one. All we have to do then is to fire the printer needles so that each needle striking the paper corresponds to a binary 1 in the appropriate byte. And now we run into the second snag. In high resolution mode, the Epson printer will print eight dots at a time, in other words a byte, but as a vertical slice not a horizontal one as on the screen. Figure 2 shows the situation. What we have to do is to take the first eight bytes of screen memory, and rearrange them for the printer so that it can print eight bytes too, but vertically.

The machine code part of the program does this by using a

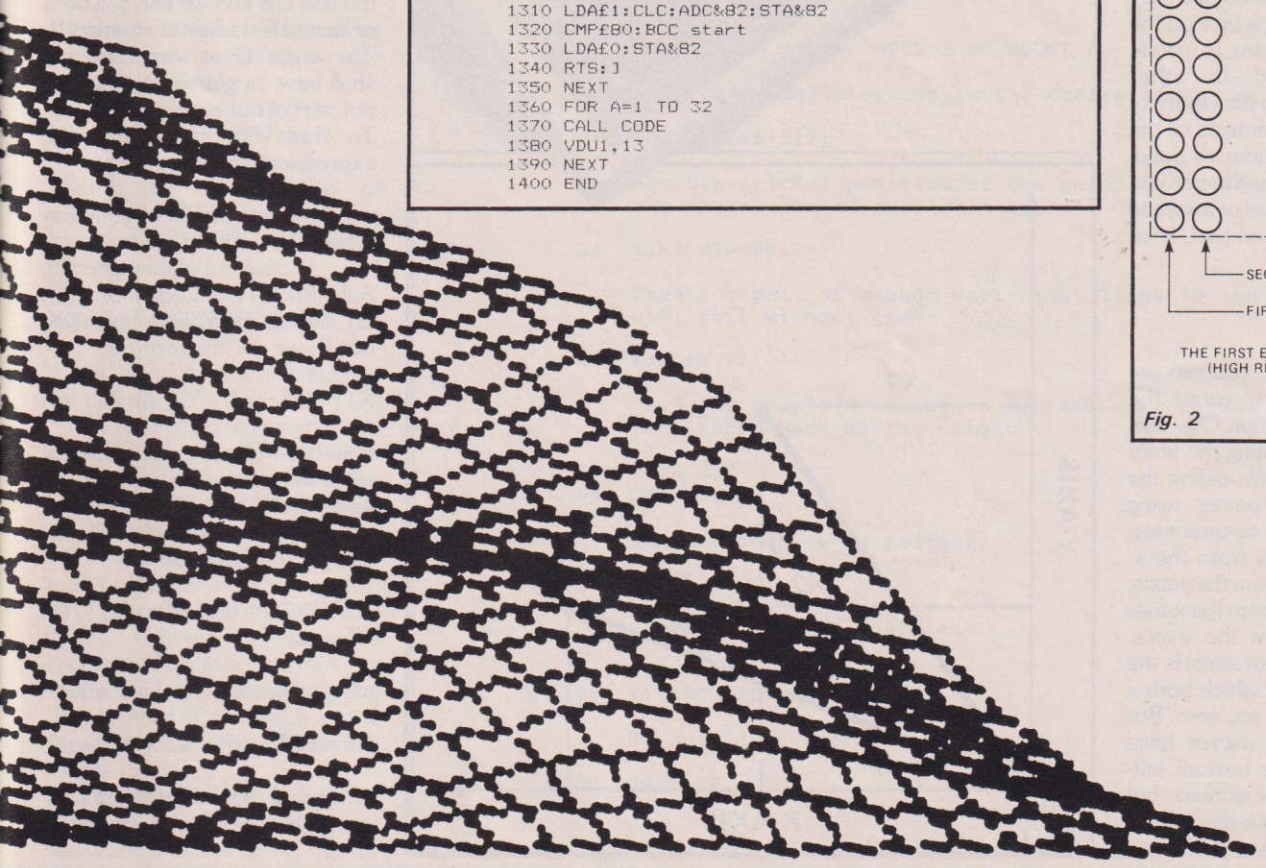


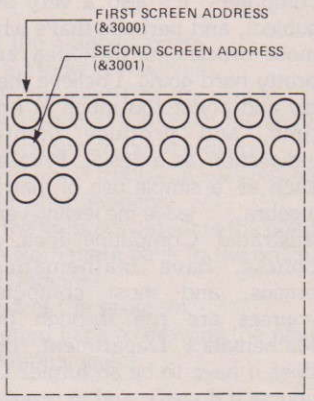
Is it a giant slug, or is it a pair of lips? No — it's just one of the many patterns you can create and save using this quick routine.

'rotate' instruction (ROL). It effectively takes the bits from the first byte of screen memory and puts them one by one into eight other memory locations, in the area reserved for machine code programs (&70 to &8F Hex). It then repeats the process for the next seven bytes needed by the printer, so they are printed out. The program then goes on to the next block of eight bytes and so on, until all 80 character positions in the line have been covered. It then drops out of the machine code section into BASIC, and lines 420 to 450 allow you to print any number of lines up to the full screen. So far so good, but the spacing between lines has to be changed to get a proper graphics picture. This is done by line 19 which uses the Epson codes.

PROGRAM LISTING

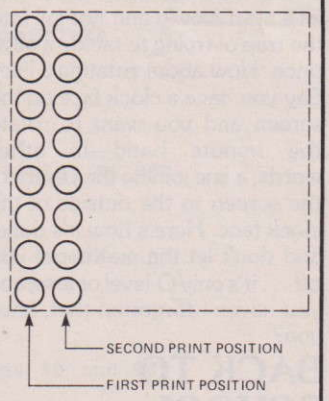
```
1000 REM Screen Dump by Norman.W.Fox
1005 REM for MODE 0 and EPSON MX 80
1010 ?(&80)=0:?(&82)=0:?(&83)=0
1020 ?(&81)=&30
1030 VDU1,27,1,65,1,7
1040 DIM CODE 500
1050 P%=CODE
1060 FOR I=0 TO 1
1070[OPT 2
1080 LDA£1:JSR&FFEE:LDA£27:JSR&FFEE
1090 LDA£1:JSR&FFEE:LDA£76:JSR&FFEE
1100 LDA£1:JSR&FFEE:LDA£128:JSR&FFEE
1110 LDA£1:JSR&FFEE:LDA£2:JSR&FFEE
1120 LDX£0:LDY£0
1130 .start LDA (&80),Y
1140 .loop CLC
1150 ROL A
1160 ROL &70,X
1170 INX
1180 CPX £7
1190 BNE loop
1200 LDX £0
1210 INY
1220 CPY £8
1230 BNE start
1240 LDX £0
1250 .char LDA£1:JSR&FFEE:LDA &70,X:JSR&FFEE
1260 INX:CPX £8
1270 BNE char
1280 LDX£0
1290 LDA£8:CLC:ADC&80:STA&80
1300 LDA£0:ADC&81:STA&81
1310 LDA£1:CLC:ADC&82:STA&82
1320 CMP£80:BCC start
1330 LDA£0:STA&82
1340 RTS:]
1350 NEXT
1360 FOR A=1 TO 32
1370 CALL CODE
1380 VDU1,13
1390 NEXT
1400 END
```





THE FIRST CHARACTER BLOCK ON THE SCREEN (MODE 0)

Fig. 1



THE FIRST EPSON PRINTING BLOCK (HIGH RESOLUTION MODE)

Fig. 2

Going Round in Circles

N Fox

Watching graphics move on-screen is fascinating. You can make them with this program.

Computer graphics is a fascinating area for home computers. It's also a very big subject, and perhaps that's why most articles in this area are pretty hard going. I believe they try and cover too much at one time, and produce mental indigestion as a result. Phrases such as 'a simple use of matrix algebra . . .' leave me feeling very frustrated. Computing does, of course, have mathematical origins, and most computer courses are run through the Mathematics Department, but does it have to be so turgid? Of course it doesn't, and to prove it let's look at the problem of rotating an object on the screen.

A program to rotate a three-dimensional object on the screen will do wonders for your standing in the local computer club. But let's start slowly and not fall into the trap of trying to tackle it all at once. How about rotating a line? Say you have a clock face on the screen and you want to rotate the minute hand—in other words, a line joining the centre of the screen to the outside of the clock face. Here's how it's done, and don't let the maths put you off . . . it's only O-level or less, and you haven't forgotten that, have you?

BACK TO SCHOOL

If you look at Fig. 1, you can see that we are going to rotate the line OA to the position OB. This is the same as rotating the point A to the point B. We define the position of both points using simple x and y co-ordinates. Point A is X1 units from the x-axis and Y1 units from the y-axis; point B is X2 units from the x-axis and Y2 units from the y-axis. Point O (letter O not zero) is the origin, the point at which both x and y co-ordinates are zero. But where is it? Most micros have their origin in the bottom left-hand corner of the screen, but then again some have it in the top left-hand corner. And some can move it around (the BBC Micro, for example). For the moment, I'm going to assume that the origin is in the bottom left-hand corner—we can add the complications later!

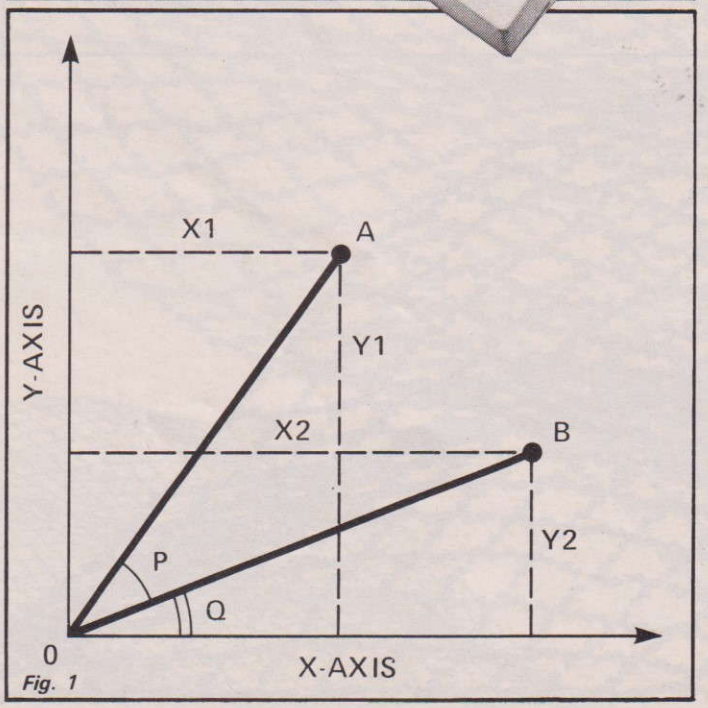


Fig. 1

So the problem is this: knowing the position of A (in other words its x and y co-ordinates) we want to write a program to rotate it to B, through an angle which I've called P. We need a formula which gives X2 and Y2 in terms of X1 and Y1, and the angle P. Now for the maths! Remembering your sines and cosines, we can write down four equations straight away:

sin Q = Y2/OB . . . (1)
cos Q = X2/OB . . . (2)

sin (P+Q) = Y1/OA . . . (3)
cos (P+Q) = X1/OA . . . (4)

But since we are talking about *rotations*, the line OA is equal to the line OB and we can call both of them the radius of rotation R. The angle Q is something we shall have to get rid of, since it's not part of our required solution. To start with there's a useful expression which says that:

sin (P+Q) = sin P*cos Q + cos P*sin Q . . . (5)

Substituting from equations (1), (2) and (3) in (5) and cancelling out R, we get the formula:

X1 = X2*cos P – Y2*sin P . . . (6)

Now this is not exactly helpful since we want a formula for X2 in terms of X1 and Y1 and not the other way round! Incidentally, by using the expression:

cos (P+Q) = cos P*cos Q – sin P*sin Q

we get the additional formula:

Y1 = Y2*cos P + X2*sin P . . .(7)

A typical bit of 'mathemagic' comes in here. If we rearrange (6) to get:

LINE DESCRIPTION

5 -10	Clear the screen and set the graphics mode. Use Mode 5 if you have a Model A machine.
20	Sets the graphics origin to the centre of the screen.
30	Sets the initial position of a point at 0,400 and the angle rotated per cycle of the program to be one degree.
40	This is a pseudo-variable to avoid trouble in lines 80 and 90. Try it without and see what happens.
50 - 60	These instructions move the graphics cursor to the point x, y and then draw a line from there back to the origin.
70	Converts D degrees to T radians.
80 - 90	The actual calculating part of the program.
100	The line to send the program back to drawing the next line—thus, creating a rotational effect.

X2 = X1/cos P + Y2*sin P/cos P

and substitute this in (7), we get:

Y1 = Y2*cos P + (X1/cos P + Y2*sin P/cos P)*sin P

Although this looks rather fearsome, it simplifies to the important formula:

Y2 = Y1*cos P - X1*sin P

Using the same kind of arguments and arithmetic, you can now deduce the other corresponding formula:

X2 = Y1*sin P + X1*cos P

Great! we've done it now how do we use it?

The first thing to remember is that in BASIC, the SIN and COS instructions need to have the angle shown in radians (in a full circle of 360 degrees there are 2*PI radians, ie 6.284 radians!). I much prefer to work in degrees, so I use the instruction RAD which converts degrees to radians. Here's a program to produce a continuously rotating line, centred in the middle of the screen it could form the basis of a simple clock.

However, before typing in the program, here is a brief breakdown of the listing.

PROGRAM LISTING

```
        First clear the screen and set the appropriate
        graphics mode

   5  CLS
  10  MODE 0:REM ** USE MODE 5 ON MODEL A

        Set the graphics origin to the centre of the screen

  20  VDU 29,640;512;

        Set the initial position of the point to 0,400 and
        the angle rotated per cycle to be 1 degree

  30  XI=0:YI=400:D=1

        Create a pair of pseudo variables, lines 80 and 90
        will fail without them

  40  X=XI:Y=YI

        Move the graphics cursor to x,y and draw a line
        from there back to the origin

  50  MOVE X,Y
  60  DRAW 0,0

        Convert degrees to radians

  70  T=RAD(D)

        Compute the next position

  80  XI=Y*SIN(T)+X*COS(T)
  90  YI=Y*COS(T)-X*SIN(T)

        Go and do it again

 100  GOTO 40
```

Beginners Bar Charts

G W Gallagher

One of the most helpful things a computer can do is translate numerical information into a pictorial representation. Computer novices will benefit from this educational article.

There seem to be two different approaches to the problem of providing illustrative graphs and charts. The first method is to write into your program procedures which will cover any possible combination of data, thus giving rise to a program which appears to be complicated, and is difficult to digest. In general, we probably use bars charts in a restricted way, according to the interests of the topic and the students.

The second method is to start with the simplest possible chart of the type required, and then, when it is running, expand parts of the program to meet additional requirements. This is the method used here to draw a bar chart, starting with the simplest possible version.

THE BASIC IDEA

Consider a set of 12 values, one for each month of the year. What the values represent does not affect the design of the program, but rather the refinements to the program later.

To begin with, the values may be placed into the program in a data statement, so that the maximum value needed is known, and can be used to judge the scales. In the first version, a maximum value of 400 was used, and scales drawn accordingly.

The first practical considerations are the size of the screen and the mode used. For this chart, MODE 0 is used. The colour is restricted (2 only), but the 80 text spaces per line allow the use of JAN, FEB, etc, along the axis, without the appearance of crowding.

SETTING UP THE SCREEN

The origin of co-ordinates is taken at 100,100, and the lines for the axes drawn from 100,1000 to 100,100 and from 100,100 to 1270,100 (Line 1010). Relating the text scale to the graphics scale, 96 points horizontally on the graphics screen is equivalent to six text spaces in MODE 0. It is convenient to take the centres of the bars at 96 points apart, as in Fig. 1. To mark these positions, short vertical lines are drawn (line 1020)

On the vertical scale, 192 points are used for each 100 value; 192 being used as a multiple of 32 to keep an easy relationship between text and graphics scales. This time, horizontal lines are used to mark the 100s (line 1030). See Fig. 1.

IN THE BALANCE

The scales on the axes are most easily added if the VDU5 instruction is included in the program. The VDU5 code allows text to be written at the graphics cursor. To use it, move to the appropriate position on the graphics screen and print the required message. See Fig. 2.

To draw the bars (Fig. 1) each bar is placed centrally on the plotted month, and has a width of 25 points on either side of the marker. This leaves a good separation (46 points) between bars.

The height of the bar is the value stored in the array multiplied by 1.92, which was the scaling factor. X1 and X2 are the base points on either side of the central marker; Y1 is the lowest point, and Y2 the highest point. See Fig. 3.

A simple title, 'Monthly Figures', is placed at the top of the screen. If a more descriptive title is required, place it in line 1160, remembering that too long a title will probably overlap the bars. See Fig. 4.

IN THE MAIN

The main program is very short. Line 10 sets the mode and

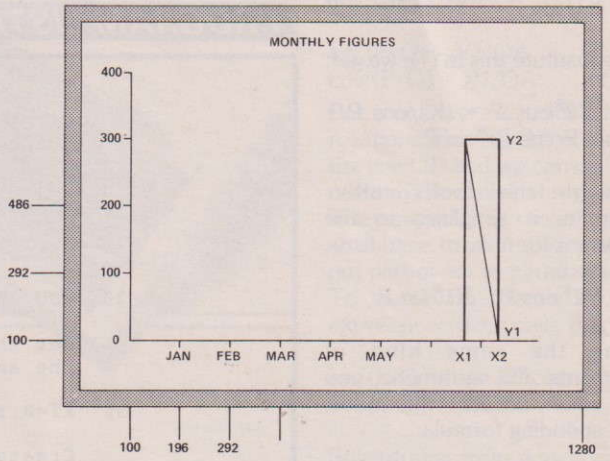


Fig. 1

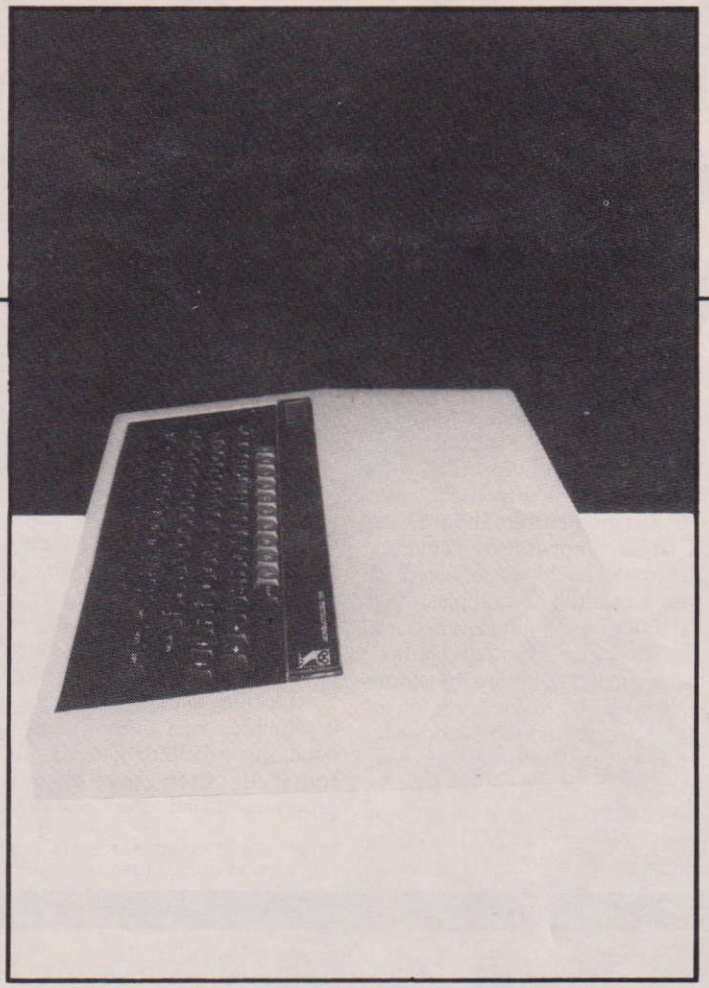
```
1000 DEFPROCAXES
1010 MOVE100,1000:DRAW100,100:DRAW1270,100
1020 FORI=1 TO 12:MOVE100+I*96,99:DRAW100+I*96,95:NEXT I
1030 FORI=1 TO 4:MOVE95,100+192*I:DRAW99,100+192*I:NEXT
I
1040 MOVE128,32:PRINT"   JAN   FEB   MAR   APR   MAY   J
UN   JUL   AUG   SEP   OCT   NOV   DEC"
1050 MOVE32,868:PRINT"400":MOVE32,676:PRINT"300":MOVE32,
484:PRINT"200":MOVE32,292:PRINT"100":MOVE32,100:PRINT"0"

1060 ENDPROC
```

Fig. 2

```
1100 DEFPROCBARS
1110 Y1=100:Y2=100+Y(I)*1.92
1120 X1=100+I*96-25:X2=100+I*96+25
1130 MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
1140 ENDPROC
```

Fig. 3

```
1150 DEFPROCTITLE
1160 MOVE160,980:PRINT"MONTHLY FIGURES"
1170 ENDPROC
```



dimensions the array ready for the values.

Line 30 reads in the data from line 800. A second set of data could be included at a different line, say 810, and the restore command altered accordingly. See Fig. 5.

Line 40 sets up the screen.

Lines 50-70 draw each bar in turn.

Line 80 adds the title.

The program is now complete, and will draw this simple bar chart quickly on the screen. When it has been run, the command VDU4 will return the cursors to their normal behaviour.

A NATURAL EXTENSION?

Having completed a simple bar chart, the next question is, how can it be extended? If the idea of 'Monthly figures' is retained, with months on the horizontal axis, there are three extensions which follow quite straightforwardly.

1. Figures could be input from the keyboard, without restriction to their value. (Above 200 the scale begins to look crowded!)
2. Using values for more than one year.
3. Using different colours to show different years, which requires a change of mode.

Mode 0 allows us six text spaces for each month on the horizontal scale. The letters in MODE 1 are larger, and there would be only three text spaces for each month. On the other hand, MODE 1 uses the background colour plus three others, which would allow the chart to be extended to three years, Choosing MODE 1, the horizontal axis uses J, F, M, etc, for the months with two spaces between the letters.

If two or three years are to be shown, side by side, on the same chart, the width of an individual bar must change. See Fig. 6.

For one year, the bar remains the same width of 25 points on each side of the marker. For two years, there are two bars, one on each side of the central marker, and each 30 points wide. For three years, there are three bars, one straddling the centre with one on each side, and each 20 points in width.

It is simpler to use three separate PROCS, one for each type of diagram; in this example these are PROCBARS1, PROCBARS2 and PROCBARS3. The method for drawing the bars is the same as in the first program, the only extra being the addition of colour.

GCOL0,1 - Draws the bar for the first year in logical Colour 1, which is red.

GCOL0,2 - Draws the bar for the second year in logical Colour 2, which is yellow.

GCOL0,3 - Draws the bar for a third year in logical Colour 3, which is white.

If the colours do not appeal to you, use the VDU19 code to change the logical colours before the bars are drawn.

GOING UP

If the values to be used are not restricted, then there must be a variable scaling factor on the vertical axis. Once the values have been placed in the array, Y(36), a check is made to find the maximum value (MAX) (lines 70-75).

For the top value on the scale, a multiple of 100 would give a neater and more easily readable chart. For that purpose, find the nearest 100 above MAX.

MAX = INT ((MAX/100) + 1) 100

```
  5 REM...BARCHART1...G.W.GALLAGHER...1983
 10 MODE0:DIMY(12)
 20 VDU5
 30 RESTORE800:FORI=1TO12:READY(I):NEXT I
 40 PROCAXES
 50 FORI=1 TO 12
 60   PROCBARS
 70   NEXT I
 80 PROCTITLE
800 DATA180,210,190,220,250,280,245,289,310,295,330,360
990 END
```

Fig. 4

Fig. 6.

```
85  PRINT "Distances on the vertical scale ?":
    INPUT D
```

Fig. 7.

CONTINUED OVER

Using the first example, where 192 points represents a distance of 100 on the vertical scale:

YSCALE = 1.92 * 400/MAX

ie multiplying 1.92 by the ratio of the old maximum (400) to the new MAX. Obviously, this line (80) could be simplified to 768/MAX, but it has been left in its present form to illustrate its derivation.

Line 1030 indicates the alterations to the markings on the vertical axis, and line 1040 the horizontal axis. In the case where the maximum value is 200 or less, this would mark only the '200', so an extra line (1070) marks the line in multiples of 50 in this situation.

Since the vertical scale is marked in mutliples of 200 (except in the special case mentioned above) the axis begins to look crowded when MAX is greater than about 2,000. If larger numbers are required, the scale could be adjusted so that it marked in multiples of 50 or 100, or even larger numbers.

The title 'Monthly figures', has been left as before, but some indication is needed of which years are plotted. This is done by printing the year on the top line, in the same colour as that in which the bars were plotted for that year (PROCTITLE and PROCHEADING).

WHAT ELSE?

Aa an example of other possibilities which this program could be adapted for, here are a few suggestions. The chart could be extended to more than three years, but the bars become rather thin. There would still only be four colours available including the background, whereas a change of mode, to MODE 2 would give the extra colours, but also very large print.

If a chart with just a few bars is required, then either program could be adapted. If colour is required, then the second program, using the one-year option with the width of the bars altered, would give the best results. If, however, the chart is required to show longer descriptions than the three-letter months, then the first program would be easier to use.

It is possible to add two or three lines in the main program (even one line would do it!) to feed in from the keyboard the distance apart of the markings ont he vertical scale.

85 PRINT "Distances on the vertical scale?": INPUT D

This would mean altering line 1060 to:

```
1060  FOR I=0 TO MAX STEP D:MOVE 32,100+YSCALE*I:PRINT I:
      NEXT:GOTO 1090
```

PROGRAM LISTING

```
    5 REM...BARCHART 2...G.W.GALLAGHER...1983
   10 DIMY(36)
   20 CLS
   30 PRINT"Do you wish to enter values for 1,2 or  3 yea
rs?"
   40 INPUTNY:IF (NY-1)*(NY-2)*(NY-3)THEN30
   50 N=NY*12
   60 FOR I=1 TO N:PRINTI;" A value,please":INPUTY(I):NEXT
   70 MAX=0:FORI=1TON:IF Y(I)<=MAX THEN 75 ELSE MAX=Y(I)
   75   NEXT
   80 MAX=(INT(MAX/100)+1)*100
   90 YSCALE=1.92*400/MAX
  100 PROCTITLE
  110 MODE1
  120 VDU5
  140 PROCAXES
  150 PROCBARS
  200 PROCHEADING
  990 END
 1000 DEFPROCAXES
 1010 MOVE100,1000:DRAW100,100:DRAW1270,100
 1020 FORI=1 TO 12:MOVE100+I*96,99:DRAW100+I*96,95:NEXT I
 1030 FORI=100 TO MAX STEP 100:MOVE95,100+YSCALE*I:DRAW99
,100+YSCALE*I:NEXT I
 1040 MOVE128,32:PRINT"  J  F  M  A  M  J  J  A  S  O  N
 D "
 1050 IF MAX <200 OR MAX=200 THEN 1070 ELSE 1060
 1060 FOR I=0 TO MAX STEP 200:MOVE0,100+YSCALE*I:PRINT;I:
NEXT :GOTO 1090
 1070 FOR I=0 TO MAX STEP 50:MOVE0,100+YSCALE*I:PRINT;I:N
EXT
 1090 ENDPROC
 1100 DEFPROCBARS1
 1105 GCOL0,1:FOR I=1 TO 12
 1110   Y1=100:Y2=100+Y(I)*YSCALE
 1120   X1=100+I*96-25:X2=100+I*96+25
 1130   MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
 1135   NEXT
 1140 ENDPROC
 1200 DEFPROCBARS2
 1210 GCOL0,1
 1215 FORI=1TO12
 1240   Y1=100:Y2=100+Y(I)*YSCALE
 1250   X1=100-15+I*96-15:X2=100+I*96
 1260   MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
 1270   NEXT
 1275 GCOL0,2
 1280 FORI=13TO24
 1290   Y1=100:Y2=100+Y(I)*YSCALE
 1300   X1=100+15+(I-12)*96-15:X2=100+(I-12)*96+15+15
 1310   MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
 1320   NEXT
 1350 ENDPROC
 1400 DEFPROCBARS
 1410 IF NY=1 THEN PROCBARS1:GOTO1450
 1420 IF NY=2 THEN PROCBARS2:GOTO1450
 1430 IF NY=3 THEN PROCBARS3
 1450 END
 1500 DEFPROCBARS3
 1510 GCOL0,1
 1520 FORI=1TO12
 1530   Y1=100:Y2=100+Y(I)*YSCALE
 1540   X1=100-30+I*96-10:X2=100-30+10+I*96
 1550   MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
 1560   NEXT
 1565 GCOL0,2
 1570 FORI=13TO24
 1580   Y1=100:Y2=100+Y(I)*YSCALE
 1590   X1=100+(I-12)*96-10:X2=100+10+(I-12)*96
 1600   MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
 1610   NEXT
 1620 GCOL0,3
 1630 FORI=25TO36
 1640   Y1=100:Y2=100+Y(I)*YSCALE
 1650   X1=100+(I-24)*96+10:X2=100+30+(I-24)*96
 1660   MOVEX1,Y1:MOVEX2,Y1:PLOT85,X1,Y2:PLOT85,X2,Y2
 1670   NEXT
 1680 ENDPROC
 1700 DEFPROCTITLE
 1710 PRINT"Year 1 is (e.g.)1980 ":INPUT Y1$
 1720 IF NY=1THEN1770ELSE 1730
 1730 PRINT"Year 2 is (e.g.)1980 ":INPUT Y2$
 1740 IF NY=2THEN1780ELSE 1750
 1750 PRINT"Year 3 is (e.g.)1980 ":INPUT Y3$:GOTO 1790
 1770 Y2$=" "
 1780 Y3$=" "
 1790 ENDPROC
 1800 DEFPROCHEADING
 1810 GCOL0,3:MOVE 32,960:PRINT"MONTHLY FIGURES"
 1820 GCOL0,1:MOVE600,960:PRINTY1$
 1830 GCOL0,2:MOVE800,960:PRINTY2$
 1840 GCOL0,3:MOVE1000,960:PRINTY3$
 1850 ENDPROC
```

Software Listings

Want a program for your BBC Micro. Look no further than our listings to make your choice.

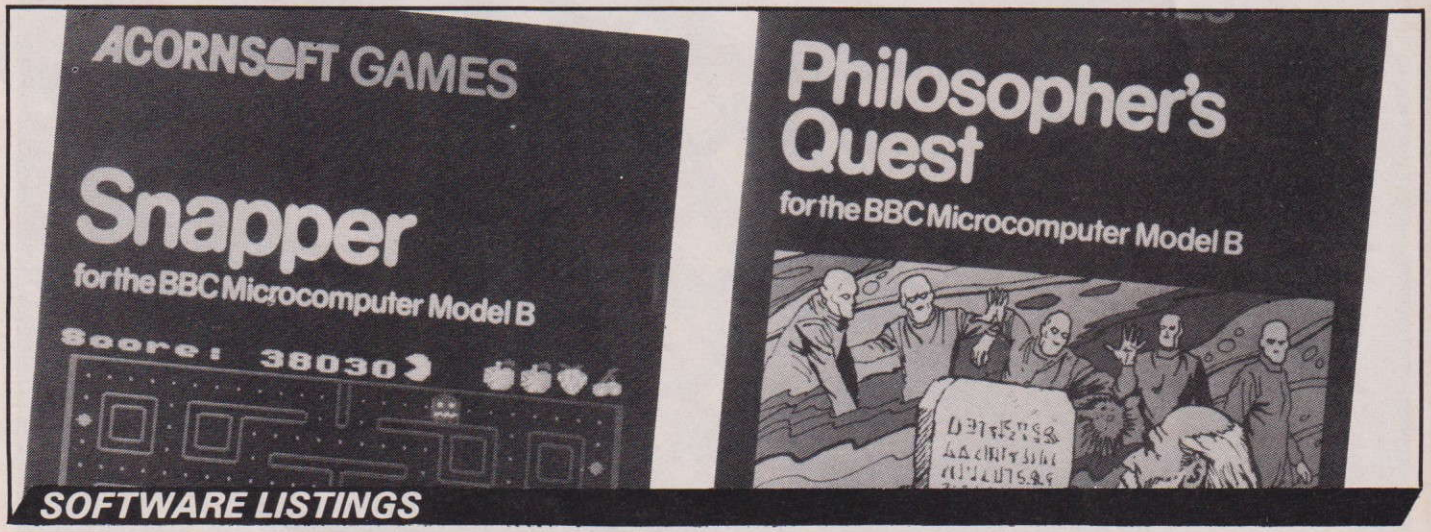
Finding and choosing the correct software for your needs is a daunting task indeed, whether you are looking for a word processing package or for a new game to test your alien-destroying, treasure-seeking, path-finding talents!

Often you can be put off even looking through the pages of advertisements which tempt you with vivid descriptions of the amazing graphics and sound effects of the game being offered, you sit there with pen poised above cheque book and your eye catches the small print that tells you that the game will run on just about every available machine except the one sitting beside you!

But BBC Micro owners despair no longer — help is at hand in the following pages. We have put together as comprehensive a list as possible of the software available for the BBC Micro. In order to fit in as many as possible we have had to use codes in some columns. The title of the software, the memory required to run it, the company that produces it, whether it is tape/cartridge or disc, the supplier and the price, are given for each piece of software listed. The codes used are:

Code	Explanation
Gm	Game
Bs	Business Routine
Ut	Utility (ie programming aid)
Do	Domestic
Ed	Educational
C	Cassette

As you are probably aware new software is surfacing all the time so don't assume that there is no such item as the one you are looking for if it is not included in this list. Also, remember that games in particular that at present run only on a machine other than the BBC Micro may well soon appear in a BBC version. If you are aware of a piece of software that is not listed here, whether you are a user or a producer, feel free to let us know.



SOFTWARE LISTINGS

Title	Type	Manufacturers	Memory	Software Supplier	Price
Action of the Heart	Ed	Garland Comp.	32K	C JX	£11.76
Adventure	Gm	Micro Power	32K	C GK	£6.95
Adventure	GM	Program Direct	32K	C NP	£5.99
Adventure Quest	Gm	Level 9 Computing	32K	C CU	£9.90
Algebraic Manipulation	Ut	Acornsoft	16K	C AL,GA	£9.95
Algebraic Manipulation	Ut	Acornsoft	32K	C AL,GA	£9.95
Alien Destroyers	Gm	Micro Power	32K	C GK	£6.95
Alien Dropout	Gm	Superior Software	32K	C KH	£6.50
Arcade Action	Gm	Acornsoft	16K	C GA	£11.90
Arrow of Death (1)	Gm	Digital Fantasia	16K	C IT	£6.95
Arrow of Death (2)	Gm	Digital Fantasia	16K	C NT	£8.95
Airlift	Gm	Bug Byte	32K	C KP	£5.50
Angle(4)	Ed	Chalksoft	32K	C KT	£6.95
Append It	Ut	Aztec S/W	16K	C IB	£3.00
Asteroids/Frong	Gm	Aardvark Software	16K	C IU	£4.00
Asteroid Belt	Gm	Electronics Appplied	32K	C IF	£11.50
Asteroid Belt	Gm	Computer Concepts	16K	C GJ	£7.80
Astro Navigator	Gm	Micro Power	32K	C GK, NR	£6.95
Atlantis	Gm	IJK Software	32K	C IT	£6.95
Awari	Gm	Foilkade	16K	C NR	£5.95
Backgammon	Ut	Bug Byte	32K	C EA,KP	£8.00
Balloons	Gm	C J E Microcomputers	32K	C NV	£6.00
Basic Maths	Ed	Aztec S/W	16K	C IB	£3.00
Battlefield	Gm	Micro-Aid	32K	C IZ	£2.50
Banner	Do	Micro-Aid	16K	C IZ	£2.95
Beeb-Chase	Gm	Database Software	32K	C NU	£7.50
Beebmunch	Gm	I.J.K. S/W	32K	C IT	£5.95
Beebtrek	Gm	Software for All	16K	C KN	£7.95
BEEP-BEEB	Gm	IJK	32K	C IT	£3.95
Bounty Pirates	Gm	Aztec S/W	16K	C IB	£5.50
Break-Up	Gm	Miking S/W	32K	C KC	£3.95
Breakout	Gm	I.J.K. S/W	16K	C IT	£3.95
Breakout	Gm	Bryants S/W	32K	C HW	£3.75
Bridgeman	Gm	Bridge S/W	32K	C KJ	£7.90
Call Your Bluff	Gm	Square Software	16K	C NY	£6.00
Capitol Cities	Ed	Square Software	16K	C NY	£6.00
Carbohydrate Metabolism	Ed	Garland Comp.	32K	C JX	£18.24
Cashbook A	Do	Micro-Aid	16K	C IZ	£3.95
Cashbook B	Do	Micro-aid	16K	C IZ	£3.95

Cat & Mouse	Gm	Micro Power	16K	C GK	£4.95
Cells and Serpents/ Stockmarket	Gm	ASP Software	16K	C OD	£11.45
Cells & Serpent	Gm	Hexagon S/W	16K	C JA	£5.00
Centipede	Gm	Superior S/W	32K	C KH	£7.00
CESCIL	Ed	Eduquest	16K	C NW	£19.95
Character Builder	Ut	Davensoft	16K	C NX	£4.95
Characters	Ut	Computer Concepts	32K	C GJ	£6.67
Character Generator	Ut	MP S/W	32K	C JZ	£3.00
Character Generator	Ut	Software for All	32K	C KN	£4.95
Characters	Ut	Computer Concepts	16K	C GJ	£6.67
Characters & Envelope Definer	Ut	Electronics Applied	32K	C IF	£5.50
Chess	Gm	Bug Byte	32K	C EA	£11.50
Chess	Gm	Micro Power	32K	C GK	£6.95
Circus	Gm	Digital Fantasia	32K	C NT	£8.95
Claws	Ed	Bryants S/W	16K	C HW	£3.75
Cards	Gm	Micro-Aid	16K	C IZ	£2.95
Cobra/Robo-Swamp	Gm	Software for All	16K	C KN	£6.95
Cobra/Robo-Swamp	Gm	Software for All	32K	C KN	£6.95
Code Race	Gm	Computer Concepts	16K	C GJ	£6.67
Code Race	Ut	Computer Concepts	32K	C GJ	£6.67
Colossal Adventure	Gm	Level 9 Computing	32K	C CU	£9.90
Connect 4	Gm	Database Software	32K	C NU	£5.90
Constellation	Ed	Micro Power	32K	C GK	£5.95
Cookbook Wizardry	Do	Database Software	32K	C NU	£7.50
Cowboy Shoot-out	Gm	Micro Power	32K	C GK	£5.95
Creative Graphic Pack	Ut	Acornsoft	16K	C AL,GA	£9.95
Creative Graphics	Ut	Acornsoft	32K	C AL,GA	£9.95
Crossed Words	Ed	Aztec S/W	16K	C IB	£6.50
Data-Quiz	Ut	Bryants S/W	32K	C HW	£4.88
Database	Bs	Computercat	16K	C IJ	£11.95
Database	Bs	Software for All	16K	C KN	£9.95
Defchr	Ut	Micro-Aid	16K	C IZ	£2.95
Defender	Gm	Acornsoft	32K	C AL,GA	£9.95
Desk Diary	Bs	Acornsoft	32K	C AL,GA	£9.95
Desk Diary	Bs	Acornsoft	16K	C AL	£9.95
Devil's Causeway	Gm	Anirog Computers	16K	C 0A	£6.00
Dissassembler	Ut	Micro Power	16K	C GK	£5.95
Dissassembler	Ut	Program Direct	16K	C NP	£3.00
Dissassembler	Ut	Davansoft	16K	C NX	£5.95
Dissassembler	Ut	C J E Microcomputers	16K	C NV	£5.00
Distances	Ed	Micro-Aid	32K	C IZ	£2.95
Dissembler	Ut	Rainbow S/W	16K	C KS	£2.50
Digital X-Word Compiler	Gm	N. Darwood	16K	C JB	£6.00
DNA Replication	Ed	Garland Comp.	32K	C JX	£17.65
Dragon Rider	Gm	Salamander Software	32K	C NZ	£6.95
Dragon Quest II	Gm	Bug Byte	32K	C KP	£11.50
Dragon Quest	Gm	Bug Byte	32K	C KP	£11.50
Drawing	Ut	B.B.C.	16K	C KB	£10.00
Dungeon Adventure	Gm	Level 9 Computing	32K	C CU	£9.90
Early Learning	Ed	B.B.C.	16K	C KB	£10.00
Early Numbers	Ed	Bryants S/W	32K	C HW	£4.80
Early Warning	Gm	A&F Software	16K	C GE	£6.00
Educational (1)	Ed	Golem	16K	C OB	£8.05
Educational (2)	Ed	Golem	16K	C OB	£8.05
7 Educational Games	Gm	Micromail	32K	C OE	£5.75
Eldorado Gold	Gm	Program Power	32K	C GK	£7.99
Electric	Ed	Database Software	16K	C NU	£5.50
European Studies	Ed	Aztec S/W	32K	C IB	£6.50
Escape from Pulsar 7	Gm	Digital Fantasia	32K	C NT	£8.95
Fairytale	Gm	Molimerx	32K	C AJ	£10.06
Feasibility Experiment	Gm	Ditigal Fantasia	32K	C NT	£8.95
Filer	Bs	Micro Power	16K	C GK	£8.95
Firien Wood	Gm	MP S/W	32K	C JZ	£6.50
Flags	Ed	Micro-Aid	16K	C IZ	£2.95
Flush	Ut	Micro-Aid	16K	C IZ	£1.00
Football Pools Predictor	Do	Mayday Software	16K	C IX	£4.99
Footer	Gm	Micro Power	32K	C GK	£6.95
Frenzy	Gm	Persoft	16K	C IY	£5.75
Frogger (Machine Code)	Gm	A&F Software	32K	C GE	£8.00
Fruit Machine	Gm	Superior Software	32K	C KH	£6.50
Fruit Machine	Gm	Bug Byte	32K	C KP	£5.50
FORTH	Ut	Acornsoft	32K	C AL	£16.85
Fun Games	Gm	B.B.C.	16K	C KB	£10.00
Galactic Commander	Gm	Micro Power	32K	C GK	£6.95
Games Pack 1	Gm	Computersmith	16K	C LC	£5.50

SOFTWARE SUPPLIERS

AJ Molimerx Ltd
1 Buckhurst Road
Town Hall Square
Bexhill-on-sea
East Sussex

AL Acornsoft Ltd
4a Market Hill
Cambridge CB2 3NJ

CU Level 9 Computing
229 Hughenden Road
High Wycombe
Buckinghamshire HP13 5PG

FY Wida Software
2 Nicholas Gardens
London W5 5HY

GA Eltec Services Limited
217 Manningham Lane
Bradford BD8 7HH

GE A&F Software
83 Hyde Road
Gorton
Manchester M18 7JD

GJ Computer Concepts
16 Wayside
Chipperfield
Hertfordshire WD4 9JJ

GK Micro Power Ltd
8/ 8a Regent Street
Chapel Allerton
Leeds LS7 4PE

HW Bryants (Educational) Software
1 The Hollies
Chalcraft Lane
North Bersted
Bognor Regis PO21 5SX

IF Electronics Applied
4 Dromore Road
Carrickfergus
County Antrim BT38 7PJ

IT IJK Software
55 Fitzroy Road
Bispham
Blackpool

IU Aardvark Software
15 Queensberry Avenue
Hartlepool
Cleveland TS26 9NW

IV James Hager
7 Basset Street
Camborne
Cornwall TR14 8SW

IW Simon Hessel Software
15 Lytham Court
Cardwell Crescent
Bershire

IX Mayday Software
181 Portland Crescent
Stanmore
Middlesex HA7 1LR

IY Persoft
Freepost
Baildon
Shipley
West Yorkshire BD17 5SX

IZ Micro-Aid
25 Fore Street
Praze-an-Beeble
Camboren
Cornwall TR14 0JX

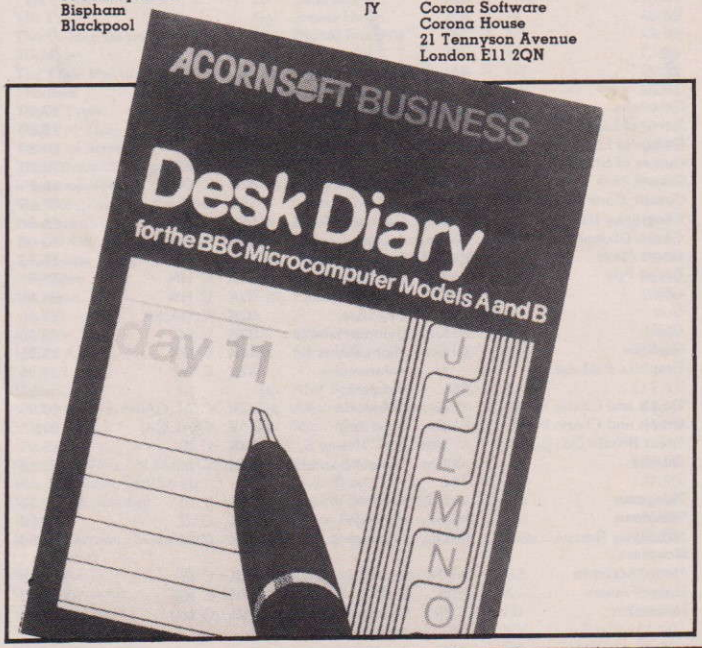
JA Hexagon Software
17 Cambridge Grove
Otley

JB N Darwood Ltd
Halfacres
Stroud
Petersfield
Hampshire GU32 3PJ

JC Futura Software
63 Lady Lane
Chelmsford
Essex CM2 OTQ

JX Garland Computing
35 Dean Hill
Plymouth PL9 9AF

JY Corona Software
Corona House
21 Tennyson Avenue
London E11 2QN



CONTINUED OVER

SOFTWARE SUPPLIERS

JZ MP Software & Services
165 Spital Road
Bromborough
Merseyside L62 2AE

KB BBC Publications
British Broadcasting Corporation
35 Marylebone High Street
London W1M 4AA

KH Superior Software
69 Leeds Road
Bramhope
Leeds

KN Software for All
72 North Street
Romford
Essex

KT Chalksoft
Lowmoor Cottage
Tonedale
Wellington
Somerset TA2 10AL

NP Program Direct
37B New Cavendish Street
London W1M 8JR

NR Foilkade
Dept PR14
66 Littledean
Yate
Bristol BS17 4UQ

NT Digital Fantasia
24 Norbreak Road
Norbreck
Blackpool

NU Database Software
97 Defoe Drive
Park Hill
Stoke-on-Trent

NV CJE Microcomputers
25 Henry Avenue
Rustington
West Sussex BN16 2PA

NW Eduquest
Thames Avenue
Windsor
Berkshire SL4 1QP

NX Davansoft
1 Delapoer Drive
Haverfordwest
Dyfed SA61 1HX

NZ Salamander Software
27 Ditchling Rise
Brighton
East Sussex BN1 4QL

OA Anirog Computers
26 Balcombe Gardens
Horley
Surrey

OB Golem Ltd
77 Qualitas
Bracknell
Berkshire RG12 4QG

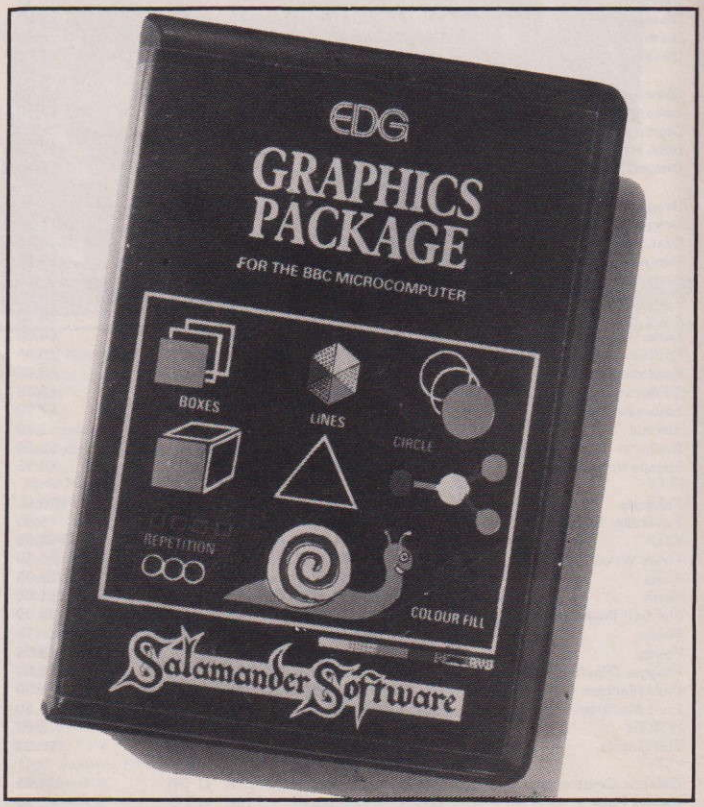
OC A Lane (Software)
12/316 Seaside
Eastbourne
East Sussex BN22 7RH

OD ASP Software
145 Charing Cross Road
London WC2H 0EE

OE Micromail
PO Box 34
Leighton Buzzard LU7 8SJ

SOFTWARE LISTINGS

Title	Type	Manufacturers	Memory	Software Supplier	Price
Galaxians	Gm	Superior S/W	32K	C KH	£7.00
Game of Logic	Ed	N. Darwood	16K	C JB	£8.00
Games of Logic and cunning	Gm	Golem	16K	C 0B	£9.20
Games of Strategy	Gm	B.B.C.	16K	C KB	£10.00
Games Pack II	Gm	Micromail	32K	C OE	£6.75
Genetic Code	Ed	Garland Comp.	32K	C JX	£17.65
Geography Italy	Ed	Corona S/W	32K	C JY	£5.00
Ghost/Diamonds	Gm	A Lane	16K	C 0C	£3.00
Ghost Maze	GM	Software for All	32K	C HN	£6.95
Grand Prix	Gm	Software for All	32K	C HN	£5.95
Golf	Gm	Bryants S/W	32K	C HW	£4.88
Golf	Gm	Bug Byte	32K	C GA,EA	£7.00
Golf	Gm	Computersmith	32K	C LC	£5.50
Gomoku	Gm	Micro Power	16K	C GK	£3.95
Graphics Package	Ut	Salamander Software	32K	C NZ	£24.95
Graph and Charts Pack	Bs	Acornsoft	32K	C AL,GA	£9.95
Graph and Charts Pack	Bs	Acornsoft	16K	C AL,GA	£9.95
Great Britain Ltd	Gm	S.W. Hessel S/W	32K	C IW	£5.95
Guzzler	Gm	Computersmith	32K	C LC	£5.50
Hangman	Gm	Aztec S/W	16K	C IB	£5.50
Hangman	Gm	MP S/W	32K	C JZ	£4.00
Helicopter Rescue/Tunnel/ Roadrace	Gm	A Lane	16K	C 0C	£4.00
Home Accounts	Do	Persoft	32K	C IY	£12.50
Home Finance	Do	B.B.C.	16K	C KB	£10.00
Hydraulics	Ed	Database Software	16K	C NU	£5.50
Inkosi	Gm	Chalksoft	32K	C KT	£5.95
Invaders	Gm	Superior S/W	32K	C KH	£7.00
Invaders	Gm	Software for All	16K	C KN	£6.95
Invaders	Gm	Hexagon S/W	16K	C JA	£6.00
Invaders	Gm	MP S/W	32K	C JZ	£6.50
Invisible Man	Ed	Chalksoft	32K	C KT	£5.95
Inheritance	Gm	S.W. Hessel S/W	32K	C IW	£5.95
Inheritance	Ed	Garland Comp.	32K	C JX	£34.70
J.R.	Gm	Software for All	32K	C KN	£6.95
Jumbles	Ed	Bryants S/W	32K	C HW	£4.88
Jumbo	Gm	Molimerx	32K	C AJ	£17.25
Junior Maths Pack	Ed	Micro Power	32K	C GK	£5.95
Katakombs	Gm	Golem	32K	C 0B	£9.20
La Princesse (French)	Ed	Aztec S/W	32K	C IB	£6.50
Landfall & Serpent	Gm	GT Software	32K	C JW	£6.50
Laser Command	Gm	Micro Power	32K	C GK	£6.95
Letters	Ed	Chalksoft	32K	C KT	£6.95
Library Dewey Classification	Ed	Aztec S/W	32K	C IB	£6.50
Link-4-Plus	Gm	ABC Software	16K	C KR	£6.95
Lisp	Ut	Acornsoft	32K	C AL,GA	£16.85
Lisp	Ut	Acornsoft	16K	C AL,GA	£16.85
Logo 2	Ut	Computer Concepts	32K	C GJ	£10.00
Lunar Lander	Gm	A&F Software	32K	C GE	£6.90
Mailing A	Bs	Micro-aid	16K	C IZ	£3.95
Mailing B	Bs	Micro-Aid	16K	C IZ	£3.95
Martians	Gm	Micro Power	32K	C GK	£5.95
Mastermind	Gm	Micro Power	16K	C GK	£3.95
Maze Invaders	Gm	Micro Power	32K	C GK	£4.95
Maze Man	Gm	C J E Microcomputers	32K	C NV	£6.00
Micro Budget	Do	Micro Power	16K	C GK	£6.95
Metrics (5)	Ed	Chalksoft	32K	C KT	£9.95
Micro Maths	Ed	LCL	16K	C KA	£24.50



Minefield	Gm	Eduquest	16K	C NW	£5.95
Master Copier	Ut	Aztec S/W	16K	C IB	£6.50
Maths Translation	Ed	Corona S/W	32K	C JY	£5.00
Micro-Derby	Gm	Bug Byte	32K	C KP	£5.50
Micro Man	Gm	Pro S/W	32K	C LD	£8.00
Middle Kingdom	Gm	Pro S/W	16K	C LD	£8.00
Minefield	Gm	A&F Software	16K	C GE	£6.00
Missile Control	Gm	C J E Microcomputers	32K	C NV	£9.00
Mission Impossible	Gm	Aztec S/W	16K	C IB	£6.50
Mixed Games	Gm	I.J.K. S/W	16K	C IT	£3.95
Model A Invaders	Gm	I.J.K. S/W	16K	C IT	£4.95
Model B Invaders	Gm	I.J.K. S/W	32K	C IT	£6.95
Money Box	Ed	Bryants S/W	32K	C HW	£4.88
Monsters	Gm	Acornsoft	32K	C AL,GA	£9.95
Monster Battles	Gm	Bryants S/W	32K	C HW	£4.80
Multifile	Bs	Bug Byte	16K	C EA	£25.00
Multiple Choice	Ed	Eduquest	32K	C NW	£25.00
Munchyman	Gm	Micro Power	16K	C GK	£5.95
Music	Do	B.B.C.	16K	C KB	£10.00
Music Maker	Gm	Rainbow S/W	16K	C KS	£3.50
Musical Numbers	Ed	Bryants S/W	32K	C HW	£4.88
Musical Number Box	Gm	Bryants S/W	16K	C HW	£3.75
Music Program	Do	Golem	32K	C OB	£6.90
Mutant Invaders	Gm	I.J.K. S/W	16K	C IT	£5.95
MX 80 Type 3 Screen Dump	Ut	Software for All	16K	C KN	£6.95
Othello	Gm	Computer Concepts	16K	C GJ	£8.95
Othello	Gm	Computer Concepts	32K	C GJ	£8.95
Othello	Gm	Computercat	16K	C IJ	£8.95
Parity	Ed	N. Darwood	16K	C JB	£6.00
Payroll	Bs	Micro-Aid	32K	C IZ	£5.95
Payroll 2	Bs	Micro-Aid	32K	C IZ	£5.95
Peeko Computer	Bs	Acornsoft	16K	C AL,GA	£9.95
Pete the Plastered Postman/	Gm	ASP Software Asteroid Lander	16K	C OD	£8.50
Pharoah's Tomb	Gm	A&F Software	32K	C GE	£8.00
Philosopher's Quest	Gm	Acornsoft	16K	C GA	£9.95
Picsave	Ut	Hexagon S/W	16K	C JA	£6.00
Painting	Ut	B.B.C.	16K	C KB	£10.00
Peeko Computer	Ut	Acornsoft	32K	C AL,GA	£9.95
Procflush	Ut	Micro-Aid	16K	C IZ	£1.00
Powerboat Race	Gm	Futura S/W	32K	C JC	£7.95
Polaris	Gm	Bug Byte	32K	C KP	£5.50
Procvar	Ut	Micro-Aid	16K	C IZ	£1.95
Princess	Gm	Aztec S/W	16K	C IB	£6.50
Punctuation	Ed	Bryants S/W	32K	C HW	£4.88
Reversi & Gomuko	Gm	Pro S/W	16K	C LD	**£8.00**
Reversi 1	Gm	Micro Power	16K	C GK	**£4.95**
Reversi 2	Gm	Micro Power	32K	C GK	**£4.95**
Road Runner	Gm	A&F Software	32K	C GE	**£6.90**
Robot	Gm	Miking S/W	32K	C KC	**£4.95**
Roulette	Gm	Micro Power	32K	C GK	**£4.95**
Searchbas	Ut	Micro-Aid	16K	C IZ	£1.95
Skwosh	Gm	Miking S/W	32K	C KC	£3.95
Seed Germination	Ed	Garland Comp.	32K	C JX	£18.82
Sequences	Ed	Chalksoft	32K	C KT	£5.95
Sheepdog Trials	Ed	Bryants S/W	32K	C HW	£4.80
Shootout	Gm	MP S/W	32K	C JZ	£5.00
Snake	Gm	Computer Concepts	32K	C GJ	£6.67
Snapper	Gm	Acornsoft	32K	C AL,GA	£9.95
Sort M/C	Ut	Micro-Aid	16K	C IZ	£1.00
Sortbas	Ut	Micro-Aid	16K	C IZ	£1.00
Soundplan	Ut	ME & P Products	16K	C KK	£10.00
Spacehawks	Gm	Computer Concepts	32K	C GJ	£7.80
Spacemaze	Gm	Micro Power	32K	C GK	£5.95
Spacefight	Gm	Miking S/W	32K	C KC	£7.95
Space Adventure	Gm	Pro S/W	32K	C LD	£8.00
Space Warp	Gm	Bug Byte	32K	C GA,EA	£11.50
Space Fighter	Gm	MP S/W	32K	C JZ	£8.50
Space Fighter	Gm	Superior S/W	32K	C KH	£7.00
Space Games Pack 1	Gm	Futura S/W	16K	C JC	£3.99
Space Games Pack 2	Gm	Futura S/W	16K	C JC	£3.99
Space Games Pack 3	Gm	Futura S/W	16K	C JC	£4.99
Space Games Pack 4	Gm	Futura S/W	16K	C JC	£4.99
Space Invaders	Gm	Pro S/W	16K	C LD	£8.00
Space Trek	Gm	Program Direct	32K	C NP	£5.99
Space Pirates	Gm	Bug Byte	16K	C KP	£8.00
Spèechparts	Ed	Bryants S/W	32K	C HW	£4.88
Squash	Gm	Aztec S/W	16K	C IB	£5.50
Star Maze	Gm	Database Software	32K	C NU	£7.50
Star Trek	Gm	Micro Power	16K	C GK	£4.95
Startrek/Candy Floss	Gm	I.J.K. S/W	16K	C IT	£5.95
Star Treck	Gm	Hexagon S/W	16K	C JA	£5.50
Statpack	Ed	Micro-Aid	32K	C IZ	£7.95
Storybuilder	Ed	Bryants S/W	32K	C HW	£4.88
Stats 1	Bs	ME & P Products	16K	C KK	£15.00
Superlife	Gm	Golem	32K	C OB	£9.20
Super Hangman	Gm	I.J.K. S/W	32K	C IT	£3.95
Swamp Monsters	Gm	M P Software	32K	C JZ	£6.50
Swoop	Gm	Micro Power	32K	C GK	£6.95
Tables Test	Ed	Bryants S/W	32K	C HW	£4.88
Tape Copy	Ut	Davansoft	16K	C NX	£7.50
Teacher's Toolkit	Ed	Wida Software	32K	C FY	£30.00
Text Processing Pack	Bs	Eduquest	32K	C NW	£10.00
The Frog	Gm	James Hager	32K	C IV	£6.50
The Golden Baton	Gm	Digital Fantasia	16K	C NT	£8.95
3D Maze	Gm	I.J.K. S/W	32K	C IT	£3.95
The Time Machine	Gm	Digital Fantashia	16K	C NT	£8.95
Timetrek	Gm	Micro Power	32K	C GK	£6.95
Touch Typist	Ed	Computercat	16K	C IJ	£9.95
Tower of Alos	Gm	A&F Software	16K	C GE	£6.90
Towns of Britain	Ed	Aztec S/W	16K	C IB	£6.50
The Computer Programme Programs Vol 2	Do	B.B.C	32K	C KB	£10.00
The Computer Programme Programs Vol 1	Do	B.B.C.	16K	C KB	£10.00
Threed OXO	Gm	Micro-Aid	16K	C IZ	£2.50
3-D Mouse Maze	Gm	Rainbow S/W	32K	C KS	£3.50
3-D Maze	Gm	Hexagon S/W	16K	C JA	£6.00
3D Maze	Gm	Earthshock S/W	16K	L KI	£3.00
Utilities	Ut	Golem	16K	C OB	£8.05
Utility A	Ut	Micro-Aid	16K	C IZ	£4.95
Valley	Gm	ASP Software	32K	C OD	£11.45
Varkman/Meanies	Gm	Aardvark Software	16K	C IU	£4.00
Vogon Attack	Gm	Micromail	32K	C OE	£6.33
Water Relations of Plant Cells	Ed	Garland Comp.	32K	C JX	£18.82
Whole Number Arithmetic	Ed	W.E. Hunt	32K	C KQ	£7.95
What's my number	Gm	Square Software	16K	C NY	**£6.00**
Where?	Ed	Micro Power	32K	C GK	**£5.95**
White Barrows Conquering Everest	Gm	ASP Software	16K	C OD	**£11.45**
Wordwise	Gm	Computer Concepts	32K	C GJ	£74.75
Word Processor	Bs	Rainbow S/W	32K	C KS	£8.00
Word Processor	Bs	Pro S/W	32K	C LD	£19.95
Wordy	Gm	Square Software	16K	C NY	**£6.00**
World Geography	Ed	Micro Power	32K	C GK	**£5.95**
Zombies	Gm	Micro Power	16K	C GK	**£3.95**

ADVERTISEMENT RATES

1. SEMI — DISPLAY

(min 2 cm)	£8.00 per cm
6-11 insertions	£7.50 per cm
12 + insertions	£7.00 per cm

or **2. LINEAGE**
35p per word. (minimum 15 words)

Closing date for the next issue is 17th May 1983.
All advertisements in this section must be prepaid.
Advertisements are accepted subject to the terms and conditions printed on the advertisement rate card (available on request)
Send your requirements to: ASP CLASSIFIED,
145 CHARING CROSS ROAD, LONDON WC2H 0EE.

SOFTWARE EDUCATIONAL

EDUCATIONAL PROGRAMS

Are you old fashioned enough to feel that ultimately children must sit down and practise arithmetic skills?

Please send for details of our top quality programs which are used in schools throughout the country and are now available for extra tuition in the home.
Ages 4 to 13 years.

COTTAGE SOFTWARE (BBC)
Heather Cottage, Selly Hill, Whitby, N. Yorkshire.

BBC 32K BBC 16K
For Young Children (1-6 yrs)
"A First Book of Micro-Rhymes"
5 traditional nursery rhymes
See the words — Hear the tunes
Watch the moving pictures
Listen to the sound effects
only **£4.95** (inc) from **Peter Gordon**
20 Despard Rd., London N19 5NW
START THEM YOUNG!!

BBC MICRO A/B

Maths 1, Graphplot, Integration, Equation Solver and colourful Piechart. Can handle most functions, a must for maths students of all levels. £4.00. Guaranteed loading. Cheques to: **A. Lane (Software), 12/316 Seaside, Eastbourne BN22 7RH.**

BBC 32K. Four pre-school programs from Bob-soft for £5.95 inc of p&p. 65 Foresthall Road NE12 0AY.

ADVERTISE YOUR EDUCATIONAL SOFTWARE IN A & B COMPUTING & GET THE BEST RESPONSE PHONE
01 437 1002 extn 282.

SOFTWARE COMPETITIONS

BUGBEAR

WIN £££s WITH SUPER DRAGSTER FOR BBC 32K

1 or 2 players. Design the ultimate DRAGSTER and win cash prizes. Superb sounds and graphics. Beware the SUPER DRAGSTER! A winner every months — it could be you. Full details with every program. **£5.95**. Also our **EASIREADER** (Pat. Pend) — THE GREATEST PROGRAMMING AID SINCE FINGERS. **£12.25** (inclusive) or SAE for details. Cheque/PO to:

BUGBEAR, Nicholson Buildings, Templetown, South Shields, Tyne and Wear NE33 5RZ.

SOFTWARE GAMES

BBC MODEL B SOFTWARE
1 West Quest
2 Super Tank
3 Space Arena
All above £5 each or all three for £10
OZARK SOFTWARE,
96 SEAL ROAD,
BRAMHALL, STOCKPORT,
CHESHIRE.

SILVER BIRCH SOFTWARE
FOR BBC MICRO GAMES

GALACTIC SEARCH — An exciting space adventure with many features and impressive displays to help you destroy the marauding Argonauts. Model B.

PIT STOP — So you think you can drive? We'll see about that! Indianapolis eat your heart out. Model A/B.

WARNING — These Programmes are addictive. To be handled with care.

The price of **£5.95** per cassette (state choice) includes p&p and prompt delivery. Orders to:

SILVERBIRCH SOFTWARE
Valtos, Newmarket Road, Burwell, Cambs CB5 0AE.

BBC MICRO A/B GAMES
Helicopter rescue, Tunnel, Roadrace, Hangman, Ghosthunter and Diamonds.
All in colour with sound **£5**.
Cheques to: **A. Lane (Software), 12/136 Seaside, Eastbourne BN22 7RH.**

EARN EXTRA CASH

High cash offers for good original programs written for SPECTRUM or BBC Micros. For rapid evaluation send tape and documentation with large SAE to:

ARROWE SOFTWARE
2 Kinloss Road, Greasby, Merseyside L49 3PS.

SOFTWARE BUSINESS

BBC MAIL ORDER agency software. All club book features (16K) Cassette £3. R. Whale, 11 Allanmead Road, Bristol BS14 9AS.

SOFTWARE APPLICATIONS

BBC DISC. Intelligent Menu driven disc by auto booting copies itself. Configures disc, easily used. Disc, program, p&p, £6.00. SAE details, CD Soft, 24 Longdean Park, Hemel Hempstead HP3 8B2.

FOR ALL YOUR CLASSIFIED REQUIREMENTS PHONE ASP LTD
01 437 1002
extn. 282

BOOKS/PUBLICATIONS

CAN YOU AFFORD TO BE WITHOUT . . .
GRAPHICS FOR THE BBC MICRO

- over 100 procedures
- Model A & B compatible
- structured error-proof listings
- works from simple to complex as a TEACH-YOURSELF manual.

Available from booksellers **£2.50** or send **£2.85** for retuan mailing to

SCHOOL SOFTWARE (B)
85 Otley Road, Harrogate, North Yorkshire HG2 0DU.

NATIONWIDE SHOPS & DEALERS

CARVELLS OF RUGBY LTD
3/7 Bank Street, Rugby CV21 2QE
THE ACORN/BBC SPECIALISTS
We also stock Cumana disc drives, Epson/Olivetti printers, Microvitec Monitors, Torch ZX disc pack + BBC Software and Books.
Phone: (0788) 65275/6
half day closing Wednesdays.

IMO COMPUTER CENTRE. 39-43 Standish St. Burnley. Tel: 54299. Open 6 days. Specialises in home and business computers. ACT Sirius/Canon/Transtec/BBC/Acorn/Atom/Oric and others and peripherals for most micros.

ACCESSORIES

The only expandable console for the BBC. The console will house disc drives, 2nd processor teletext adapter etc. With all wiring out of sight within the console. **Coming soon:** an extra bolt on module for extra expansion space. Yes, this console will grow with you. Basic console as shown only **£39.99** plus £4.00 p&p. Send cheque for **£43.99**. Matching printer stand can double for VDU stand over the micro, only **£14.99** + £2.00 p&p. For more details: **SILENT COMPUTERS, 27 Wycombe Rs., London N17 9XN. Tel: 01-801 3014. MAIL ORDER ONLY. Viewing by appointment only.**

BBC MICRO CARRY CASE
Semi-hard fibre board case in grey, 2 hinges, twin locks, internal cable compartment. Retractable handle. **£24.95** inc VAT and p&p. Send cheque/Visa/Access number to: **COMPUTER FACILITIES (1982), Glebe House, Winterton Rd., Scunthorpe DN15 0RA. Tel: 0724 863167.**

PROTECT your BBC equipment with a PVC dust cover. Computer £3.95, single disk drive, £2.95, dual disk drive £3.95, monitors, £4.50, printers from £2.95. Access/Barclaycard. Sherbourne Designs, Victory House, 8A The Rank, North Bradley, Trowbridge, Wilts. Tel: 02214 4425.

AD INDEX

HAVE YOU GOT A BBC MICRO? THEN YOU NEED:

THE NEWSLETTER OF THE INDEPENDENT NATIONAL BBC MICROCUMPUTER USERS GROUP

IN RECENT MAGAZINES:

Articles: Teletext Graphics, User Definable Characters, User Definable Keys, Sounds, *FX, Telesoftware, Line Structure and Merging, Epson Screen Dump, Pixel Power, Moving Things, Manual Review, What Printer?, Seikosha Dump, Machine Code, How To Get More Colours Out Of Your BBC Micro, Circles Galore, Conversions for RGB Inputs, How To Use Joysticks, Instant Graphics, Software Protection, String, String, String, What Monitor? Speeding Up Your Programs, Computer Conversions, Questionnaire Results, Formatting, Assembler Programming On The BBC Micro, Errors? Focus On Adventure, Diskspot, Computer Programme Review, ADC Corner, Make The Most Out Of Sound, and lots, lots more . . . . .

Programs: Labyrinth (Game), Pontoon (Game), Artillery (Game), Life (Misc.), Calendar (Misc.), Prism (Educ.), Spiral Patterns (Graphics), Bazooka (Game), 4-In-A-Row (Game), Mortgage (Misc.), Oxygen (Game), Teletext Terminal (Utility), Wordprocessor (Business), Mixer (Graphics), How Many Colours? (Graphics), Disassembler (Utility), Maths Race (Educ.), Puzzle Program (Game), TV Test Signal Generator (Utility), Alphabet Tester (Educ.), Repeat (Game), Memory Analyser (Utility), Wallball (Game), Dating (Misc.), What's The Time? (Educ.), Grand Prix (Game), Nine Dice (Game), Memory Dump (Util.) and lots, lots more . . . . .

Regular Features: Bookreview, Letters, Hardspot, Softreview (we review *at least* 8 programs per magazine from all sources), Oddspot (a different graphical program every month), Meeting Place (where we list local user groups), Contacts (spans over 1 page of names and addresses of people who want to get in touch with other users in their area), Competitions, Printereview (we have looked at the Seikosha and Amber 2400 and give full details on how to use them with screen dump programs), Seasons (a seasonal program every month), Queryspot, Special Offers (special offers and exclusive club discounts), Assembler Programming On The BBC Micro, Software Protection and lots, lots more . . . . .

WE WON'T TRY TO PUSH LASERBUG ON YOU LIKE THE OTHERS – SEND OFF FOR A SAMPLE COPY AND YOU'LL FIND THAT LASERBUG SELLS ITSELF

ALSO AVAILABLE FROM LASERBUG:

Do you want a dust cover for your BBC Micro but can't find one anywhere??? If so then look no further, LASERBUG has the answer. LASERBUG Dust Covers are specially made for us in a high quality polyester/cotton looking much better than the PVC covers you can get from some places. And the real good news? They are only £3.25 each inclusive.

If you need a cassette lead for your BBC Micro the chances are that you are about to pay between £4 and £8 for the right lead! Why bother when you can get a suitable lead from LASERBUG for just £3.00. Send for either a DIN and Remote Jack or a 3 Jack lead today.

Please supply me with: A sample copy of LASERBUG @ £1.00
A 12 Month subscription to LASERBUG @ £12.00 for 12 issues of the magazine
A 6 Month subscription to LASERBUG @ £6.00 for 6 issues of the magazine
An overseas subscription to LASERBUG @ £14.00 (Surface Mail – write for details of air mail)
. . . . . LASERBUG Dust Cover(s) @ £3.25 each
. . . . . LASERBUG Cassette Lead(s) (Please state type(s)) @ £3.00 each

I enclose a cheque/PO for £ . . . . . made payable to LASERBUG

NAME . . . . . ADDRESS . . . . .

. . . . . .

. . . . . .

Please send the completed form to: LASERBUG Dept. B, 10 Dawley Ride, Colnbrook, Slough, Berks., SL3 0QH.

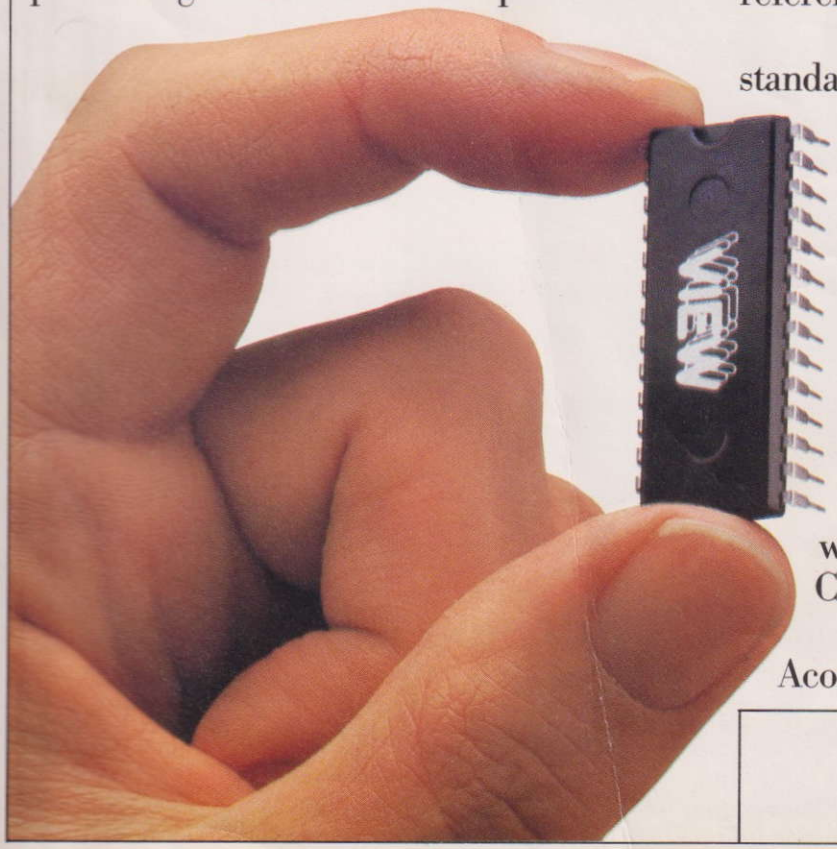
A two-minute operation turns your BBC Micro into the heart of a word processor.

VIEW is a software program from Acornsoft (the software division of Acorn Computers Ltd., who designed and built the BBC Micro) that enables you to use your BBC Micro, together with a printer, as a fully operational word processor.

View is supplied as a Rom chip that can easily be fitted to your BBC Micro by your local dealer, in a painless two-minute operation.

Then, once installed, you only have to switch on and View is operating immediately. (You can easily switch back to normal computing with a single command.)

Also included in the View package are two special books: 'Into View,' that takes you by easy stages through all the word processing commands and explains the many ways in which View can help you, and the 'View Guide,' which provides a quick reference to all View facilities.



You'll find that View is, by any standards, a thoroughly professional system, yet still surprisingly simple for the beginner to master.

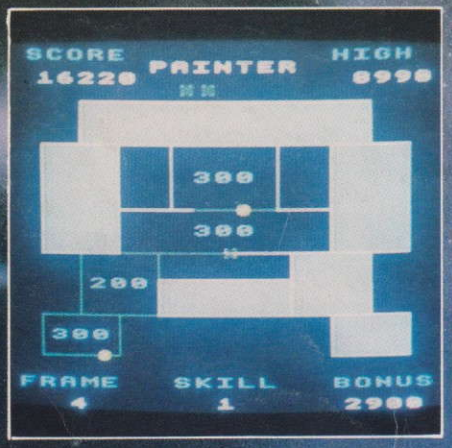
The 'Spark-Jet Printer' shown in the photograph is the ideal choice of printer for your word processing application. Extremely quiet, it offers high resolution graphics from monitor or T.V. screen and is available now from dealers.

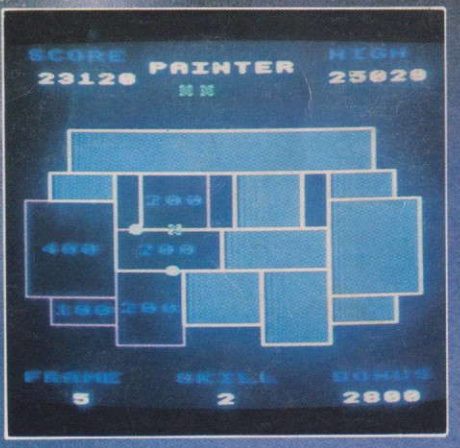
If you'd like more information, write to Acornsoft, 4a Market Hill, Cambridge CB2 3NJ.

Or, for details of your local Acornsoft dealer, phone 01-200 0200.

ACORNSOFT

32K BBC ALL O.S.'s





PAINTER Written entirely in machine code this fast arcade style game features sixteen different screen presentations and six skill levels which make the game harder as you get better. **£8.00**

PLANES Another A&F machine code special. Select either 'fast' or 'slow' and stand by your keyboard for action. Hi-score tables and brilliant graphics **£8.00**

FROGGER Also written in machine code this is a frustrating exercise in crossing the road, fording a river and trying for extra points by picking up the lady frog. Beware of the snake and don't forget the crocodile! **£8.00**

TOWER OF ALOS An excellent adventure game which has the feature of saving your character if you want to. Also for the Model A.

PHARAOH'S TOMB The interactive adventure game for the BEEB. Pick up the gold pieces, fight the spider and be scared penniless by the mummy. Work out the magic words and you are on your way. **£8.00**

LUNAR LANDER One of our old favourites and still just as difficult to get down in one piece. **£6.90**

ALL BBC PROGRAMS AVAILABLE ON DISC AT **£11.50** EACH.

AVAILABLE FROM ALL GOOD SOFTWARE HOUSES OR DIRECT FROM

A+F SOFTWARE

830 HYDE ROAD, MANCHESTER M18 7JD

061-223 6206 TELEX 667461 (Attn. A+F)